离散数学作业

Homework

尹超

中国科学院大学，北京 100049

Carter Yin

University of Chinese Academy of Sciences, Beijing 100049, China

2024.8 – 2025.1

序言

本文为笔者离散数学的作业。
望老师批评指正。

目录

Homework 1 第一章作业

1.1 习题总结

1.1.1 反证法

间接证明法（indirect proof）
直接证明法有的时候比较困难
不从前提开始、以结论结束的证明方法叫间接证明法

反证法（proof by contraposition）
归谬证明法（proof by contradiction）

反证法（proof by contraposition）
条件语句 $p \to q$ 等价于它的逆否命题 $\neg q \to \neg p$
证明当 $\neg q$ 为真时，$\neg p$ 一定为真

示例 1：
证明“如果 n 是一个整数且 $3n+2$ 是奇数，则 n 是奇数”
直接证明比较困难
假设 n 不是奇数，即 n 为偶数，则 $n = 2k$，k 为某个整数
$3n+2 = 3(2k)+2 = 6k+2 = 2(3k+1)$，即 $3n+2$ 为偶数，逆否命题为真，所以原命题也为真

示例 2：
证明“如果 n 是一个整数且 n^2 是奇数，则 n 是奇数”
直接证明比较困难：假设 n^2 是奇数，很难推导下去
假设 n 不是奇数，即 n 为偶数，则 $n = 2k$，k 为某个整数
$n^2 = (2k)^2 = 4k^2 = 2(2k^2)$，即 n^2 为偶数
即我们证明了逆否命题“如果 n 是一个偶数，则 n^2 是偶数”
由反证法，“如果 n 是一个整数且 n^2 是奇数，则 n 是奇数”

1.1.2 归谬证明法

归谬证明法（proof by contradiction）
假设我们需要证明命题 s（不一定是条件语句）
如果能够证明条件语句 $\neg s \to r$ 为真，而且 r 为矛盾式，那么 s 为真
因为 r 为矛盾式，所以条件语句 $\neg s \to r$ 等价于 $\neg s \to F$，而 $\neg s \to F$ 的逆否命题为 $T \to s$

示例 1：
证明“任意 22 天中至少有 4 天属于每星期的同一天”
假设命题为假，即“任意 22 天中至多有 3 天属于每星期的同一天”，而一个星期有 7 天，这蕴含着至多可以选出 21 天，与命题的前提“22 天”矛盾

示例 2：

证明“$\sqrt{2}$ 是无理数”
假设命题为假，即“$\sqrt{2}$ 不是无理数”，则“$\sqrt{2}$ 是有理数”
那么存在整数 p 和 q 使得 $\sqrt{2}=\frac{p}{q}$，其中 $q\neq 0$ 且 p 和 q 没有公约数
两端取平方，可得 $2=\frac{p^2}{q^2}$，即 $2q^2=p^2$
所以 p^2 为偶数，那么 p 也是偶数，存在整数 r 使得 $p=2r$
因此 $2q^2=p^2=(2r)^2=4r^2=2(2r^2)$，即 $q^2=2r^2$
所以 q^2 为偶数，那么 q 也是偶数，p 和 q 有公约数 2，矛盾

示例 3：
证明“素数的个数是无限的”
假设命题为假，即“素数的个数是有限的”
设素数的个数为 n，从小到大排列为 $p_1=2,p_2=3,\cdots,p_n$
考虑整数 $m=p_1\times p_2\times\cdots\times p_n+1$
因为 $p_1,p_2,\cdots,p_n$ 都不能整除 m，所以 m 是一个素数
这与“素数的个数为 n”矛盾
所以“素数的个数是无限的”

归谬证明法可以用于条件语句
条件语句 $p\to q$ 等价于 $\neg p\vee q$
假设 $\neg(\neg p\vee q)$ 为真，即 $p\wedge\neg q$ 为真，推导出矛盾式，则原命题为真

归谬证明法和反证法
条件语句的反证法证明可以改写为归谬证明
反证法：假设 $\neg q$ 为真，推导出 $\neg p$ 为真
归谬证明法：假设 $\neg q$ 和 p 为真，推导出 $\neg p\wedge p$ 为真

归谬证明法和直接证明法
条件语句 $p\to q$ 的直接证明可以改写为归谬证明
直接证明：假设 p 为真，推导出 q 为真
归谬证明法：假设 p 和 $\neg q$ 为真，推导出 $q\wedge\neg q$ 为真

Homework 2 第二章作业

2.1 习题总结

2.1.1 笛卡尔积

有序 n 元组（ordered n-tuple）

有时候集合中的元素顺序是很重要的

有序 n 元组是由 n 个对象组成的有序集合 $(a_1, a_2, a_3, \ldots, a_n)$，其中 a_1 为第 1 个元素，a_2 为第 2 个元素，如此类推，a_n 为第 n 个元素

两个有序 n 元组相等，当且仅当对应位置的元素都相等

序偶（ordered pair）

有序 2 元组又称序偶或有序对

序偶 (a, b) 和 (c, d) 相等，当且仅当 $a = c$ 且 $b = d$

笛卡尔积（Cartesian Product）

两个集合 A 和 B 的笛卡尔积是由序偶 (a, b) 构成的集合，其中 $a \in A, b \in B$

用记号 $A \times B$ 表示集合 A 和 B 的笛卡尔积

$A \times B = \{(a, b) \mid a \in A, b \in B\}$

示例：

- $A = \{a, b\}$，$B = \{1, 2, 3\}$
- $A \times B = \{(a, 1), (a, 2), (a, 3), (b, 1), (b, 2), (b, 3)\}$

笛卡尔积 $A \times B$ 的一个子集 R 被称为从集合 A 到 B 的关系（relation）

多个集合的笛卡尔积

n 个集合 $A_1, A_2, \ldots, A_n$ 的笛卡尔积是由有序 n 元组 $(a_1, a_2, \ldots, a_n)$ 构成的集合，其中 $a_i \in A_i, i = 1, 2, \ldots, n$

$A_1 \times A_2 \times \ldots \times A_n = \{(a_1, a_2, \ldots, a_n) \mid a_i \in A_i, i = 1, 2, \ldots, n\}$

用记号 A^2 表示 $A \times A$，A^3 表示 $A \times A \times A$，以此类推

示例：

- $A = \{0, 1\}$，$B = \{1, 2\}$，$C = \{1, 2, 3\}$
- $A \times B \times C = \{(0,1,1), (0,1,2), (0,1,3), (0,2,1), (0,2,2), (0,2,3), (1,1,1), (1,1,2), (1,1,3), (1,2,1), (1,2,2), (1,2,3$

2.1.2 可数集合

可数集合（countable set）

- 可数（countable）和不可数（uncountable）
  - 如果一个集合是有限的或者和正整数集合（$\mathbb{Z}^+$）有相同的基数，则称这个集合是可数的
  - 如果一个集合不是可数的，则称其为不可数的
  - 当一个无穷集合是可数的（即可数无穷集合），它的基数为 $\aleph_0$
  - $\aleph$ 读作阿列夫（aleph），是希伯来语（Hebrew）字母表的第一个字符
  - 记作 $S = \aleph_0$，并说集合 S 具有基数“阿列夫零（aleph null）”

证明一个集合是可数的

- 一个无穷集合是可数的，当且仅当可以把集合中的元素排列成序列（以正整数为下标）

- 这是因为从正整数到集合 S 的一一对应函数 f 可以用序列 $a_1, a_2, \ldots, a_n, \ldots$ 表示，其中 $a_1 = f(1)$, $a_2 = f(2), \ldots, a_n = f(n), \ldots$

示例 1：

- 证明正奇数集合是可数的
- 正奇数集合与正整数集合之间存在一一对应关系：$f(n) = 2n - 1$

希尔伯特大饭店（Hilbert’s Grand Hotel）

- 希尔伯特构造的一个例子，证明了某些对有限集合不可能的事情对于无穷集合变得可能
- 一个有可数无穷多个房间的大饭店，每个房间都住了客人，但是当一个新的客人来到这个大饭店，我们总是可以容纳这个新客人而不赶走旧的客人
- 当饭店的房间数量是有限时，所有房间客满的概念等价于不能再容纳新客人
- 但是房间数量是无穷时，这种等价关系不再成立

示例 2：

- 证明所有整数的集合是可数的
- 我们可以把所有整数排列为一个序列：$0, 1, -1, 2, -2, 3, -3, \ldots$
- 这是一个双射函数 $f(n) = \begin{cases} \frac{n}{2} & \text{如果} n \text{是偶数} \\ -\frac{n-1}{2} & \text{如果} n \text{是奇数} \end{cases}$

示例 3：

- 证明正有理数集合是可数的
- 每个正有理数都可以表示为两个正整数之比 $\frac{p}{q}$
- 把有理数 $\frac{p}{q}$ 排在第 q 行第 p 列
- 从左上角出发，沿着每条反对角线依次列出相应的有理数
- 遇到已经列出过的有理数，就跳过不再重复列出
- 由于所有正有理数都被列出而且只列出一次
- 因此所有正有理数排列成了一个序列

示例 4：

- 证明一个有限字母表 A 上的有限串组成的集合 S 是可数的
- 假设字母表 A 中的字符有一个字母顺序
- 我们证明集合 S 中的所有串可以排列成一个序列
  - 先列出长度为 0 的串
  - 再按字母顺序列出长度为 1 的串
  - 再按字典序列出长度为 2 的串
  - 以此类推
- 这意味着存在一个从正整数到集合 S 的一一对应，所以集合 S 是可数的

示例 5：

- 证明所有 Java 程序组成的集合是可数的
- 令 A 为所有 Java 程序中允许出现的字符构成的字母表
- 令 S 为字母表 A 上的有限串集合
- 根据前一个示例，可以把 S 中的串排列成一个序列 a_n
- 使用 Java 编译器按序列顺序，逐个检查字符串 a_i 是否一个合法的 Java 程序
- 如果字符串 a_i 通过 Java 编译器检查，则将其加入一个新的序列 b_n
- 通过以上过程，我们建立了一个从正整数到所有 Java 程序的一一对应
- 因此所有 Java 程序组成的集合是可数的

子集的基数

- 可数集合的子集是可数的
- 不可数集合的超集是不可数的
- 不可数集合一定存在可数无穷子集

交集的基数

- 可数集合与其他集合的交集是可数的
- 两个不可数集合的交集可能是可数的，也可能是不可数的

并集的基数

- 两个可数集合的并集是可数的
- 可数个可数集合的并集是可数的
- 不可数集合与其他集合的并集是不可数的

差集的基数

- 可数集合与其他集合的差集是可数的
- 不可数集合与可数集合的差集是不可数的
- 两个不可数集合的差集可能是可数的，也可能是不可数的

2.1.3 不可数集合

不可数集合（uncountable set）

Schröder-Bernstein 定理

- 如果集合 A 和 B 满足 $|A| \leq |B|$ 和 $|B| \leq |A|$，那么 $|A| = |B|$
- 如果存在一个从集合 A 到 B 的单射 f 和一个从集合 B 到 A 的单射 g，那么集合 A 和 B 是一一对应的。

示例：

- 证明 $[0,1] = [0,1)$
- 直接找一个从 $[0,1]$ 到 $[0,1)$ 的一一对应比较困难
- 因为 $[0,1) \subseteq [0,1]$，所以 $f(x) = x$ 是从 $[0,1)$ 到 $[0,1]$ 的单射
- 因为 $[0,0.5) \subseteq [0,1)$，所以 $g(x) = \frac{x}{2}$ 是从 $[0,1]$ 到 $[0,1)$ 的单射

连续统假设（Continuum Hypothesis）

- 连续统假设是关于无穷集合的基数的一个猜想
- 正整数集合的幂集与实数集具有相同的基数，即 $\mathcal{P}(\mathbb{Z}^+) = \mathbb{R} = c$
- 用 2^S 表示集合 S 的幂集的基数，因此 $c = \mathcal{P}(\mathbb{Z}^+) = 2^{\aleph_0}$
- 一个集合的基数总是小于它的幂集的基数，即 $S < \mathcal{P}(S) = 2^S$
- 连续统假设断言不存在集合 A 使得 $\aleph_0 < |A| < 2^{\aleph_0}$
- 我们将无穷集合的基数从小到大排一个顺序 $\aleph_0 < \aleph_1 < \aleph_2 < \cdots$
- 如果连续统假设成立，则有 $\aleph_1 = c = 2^{\aleph_0}$

2.1.4 函数的表示

满射函数（surjection）

映上的（onto）

双射函数（bijection）

一一对应的（one-to-one correspondence）

单射函数（injection）

是一对一的（one-to-one）

逆函数（inverse function）

反函数

复合函数（composition）

2.1.5 矩阵

0-1 矩阵（zero-one matrix）

- 如果一个矩阵的元素是 0 或 1，称这个矩阵为 0-1 矩阵
- 许多离散结构经常表示为 0-1 矩阵

布尔算术（Boolean arithmetic）

- 使用 0-1 矩阵的算法通常是基于布尔算术
- 布尔运算

$$b_1 \wedge b_2 = \begin{cases} 1 & \text{如果} b_1 = b_2 = 1 \\ 0 & \text{否则} \end{cases}, \quad b_1 \vee b_2 = \begin{cases} 1 & \text{如果} b_1 = 1 \text{或者} b_2 = 1 \\ 0 & \text{否则} \end{cases}$$

0-1 矩阵的并（join）和交（meet）

- 令 $A = [a_{ij}]$ 和 $B = [b_{ij}]$ 为 $m \times n$ 的 0-1 矩阵
- 矩阵 A 和 B 的并是一个 $m \times n$ 的 0-1 矩阵，其 i, j 项为 $a_{ij} \vee b_{ij}$
- 矩阵 A 和 B 的并，记作 $A \vee B$
- 矩阵 A 和 B 的交是一个 $m \times n$ 的 0-1 矩阵，其 i, j 项为 $a_{ij} \wedge b_{ij}$
- 矩阵 A 和 B 的交，记作 $A \wedge B$

示例：

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

$$A \vee B = \begin{bmatrix} 1 \vee 0 & 0 \vee 1 & 1 \vee 0 \\ 0 \vee 1 & 1 \vee 1 & 0 \vee 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

$$A \wedge B = \begin{bmatrix} 1 \wedge 0 & 0 \wedge 1 & 1 \wedge 0 \\ 0 \wedge 1 & 1 \wedge 1 & 0 \wedge 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

0-1 矩阵的布尔积（Boolean product）

- 令 $A = [a_{ij}]$ 为 $m \times k$ 的 0-1 矩阵，$B = [b_{ij}]$ 为 $k \times n$ 的 0-1 矩阵
- 矩阵 A 和 B 的布尔积是一个 $m \times n$ 的 0-1 矩阵，其 i, j 项为

$$c_{ij} = (a_{i1} \wedge b_{1j}) \vee (a_{i2} \wedge b_{2j}) \vee \cdots \vee (a_{ik} \wedge b_{kj})$$

- 矩阵 A 和 B 的布尔积记作 $A \bigodot B$ （符号 $\bigodot$ 有时候被用作表示 Hadamard 积）
- 矩阵 A 和 B 的布尔积等价于：先做普通矩阵乘积，再转换为 0-1 矩阵

示例：

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

$$A \bigodot B = \begin{bmatrix} (1 \wedge 1) \vee (0 \wedge 0) & (1 \wedge 1) \vee (0 \wedge 1) & (1 \wedge 0) \vee (0 \wedge 1) \\ (0 \wedge 1) \vee (1 \wedge 0) & (0 \wedge 1) \vee (1 \wedge 1) & (0 \wedge 0) \vee (1 \wedge 1) \\ (1 \wedge 1) \vee (0 \wedge 0) & (1 \wedge 1) \vee (0 \wedge 1) & (1 \wedge 0) \vee (0 \wedge 1) \end{bmatrix} = \begin{bmatrix} 1 \vee 0 & 1 \vee 0 & 0 \vee 0 \\ 0 \vee 0 & 0 \vee 1 & 0 \vee 1 \\ 1 \vee 0 & 1 \vee 0 & 0 \vee 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

0-1 矩阵的布尔幂（Boolean power）

- 令 A 为一个 $n \times n$ 的 0-1 方阵
- A 的 r 次布尔幂是 r 个 A 的布尔积，记作 $A^{[r]}$
- $A^{[0]} = I_n$
- $A^{[r]} = A \circ A^{[r-1]}, \quad r \geq 1$
- 布尔幂是良定义的，因为布尔积满足结合律

示例：

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

$$A^{[2]} = A \bigodot A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}, \quad A^{[3]} = A \bigodot A^{[2]} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$A^{[4]} = A \bigodot A^{[3]} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$A^{[5]} = A \bigodot A^{[4]} =$

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$A^{[r]} = A^{[5]}$，$r \geq 5$

Homework 3 第三章作业

3.1 习题 3.1.2

判断下列过程具有和缺乏在正文中（算法 1 之后）所描述的哪些算法特征。

a) procedure double(n: 正整数)

```
while n > 0
    n := 2 * n
```

b) procedure divide(n: 正整数)

```
while n   0
    m := 1 / n
    n := n - 1
```

c) procedure sum(n: 正整数)

```
sum := 0
while i < 10
    sum := sum + i
```

d) procedure choose(a, b: 整数)

```
x := a 或 b
```

算法的性质

- 输入（input）：算法有一个来自指定集合的输入值
- 输出（output）：对每组输入，算法从指定集合中产生输出值，即问题的解
- 明确性（definiteness）：算法的每个步骤必须是准确定义的
- 正确性（correctness）：对每组输入，算法都应该产生正确的输出值
- 有限性（finiteness）：对任意输入，算法都应该在有限（可能很多）步内产生期望的输出
- 有效性（effectiveness）：算法的每一步都应该能在有限时间内准确完成
- 一般性（generality）：算法应该可以应用于期望形式的所有问题，而不仅仅是某些特定的输入值

解. a) procedure double(n: 正整数)

```
while n > 0
    n := 2 * n
```

– 输入：正整数 n
– 输出：n 的倍数
– 明确性：明确

- 正确性：正确
- 有限性：有限
- 有效性：有效
- 一般性：一般

b) procedure divide(n: 正整数)

```
while n  0
    m := 1 / n
    n := n - 1
```

- 输入：正整数 n
- 输出：n 的倒数
- 明确性：明确
- 正确性：不正确，$n = 0$ 时除法不成立
- 有限性：有限
- 有效性：有效
- 一般性：一般

c) procedure sum(n: 正整数)

```
sum := 0
while i < 10
    sum := sum + i
```

- 输入：正整数 n
- 输出：1 到 10 的和
- 明确性：明确
- 正确性：不正确，未定义 i
- 有限性：有限
- 有效性：有效
- 一般性：一般

d) procedure choose(a, b: 整数)

```
x := a 或 b
```

- 输入：整数 a, b
- 输出：a 或 b
- 明确性：明确
- 正确性：正确
- 有限性：有限
- 有效性：有效
- 一般性：一般
- 说明：x 为 a 或 b 的值

□

3.2 习题 3.1.42

用插入排序来排序 d, f, k, m, a, b 的列表，说明在每一步所获得的列表。

(1) 初始列表：d, f, k, m, a, b

(2) 第一步：将 f 插入到 d 之后，列表保持不变：d, f, k, m, a, b

(3) 第二步：将 k 插入到 f 之后，列表保持不变：d, f, k, m, a, b

(4) 第三步：将 m 插入到 k 之后，列表保持不变：d, f, k, m, a, b

(5) 第四步：将 a 插入到 d 之前，列表变为：a, d, f, k, m, b

(6) 第五步：将 b 插入到 a 之后，列表变为：a, b, d, f, k, m

最终排序结果为：a, b, d, f, k, m

3.3 习题 3.1.62

62. 证明在解决报告厅安排一组演讲 (如例 7 所示) 的贪婪算法中，如果在每一步都选择一个与其他演讲冲突最少的演讲，则不一定产生最优解。

- 讲座 A：开始时间 = 1，结束时间 = 4
- 讲座 B：开始时间 = 3，结束时间 = 5
- 讲座 C：开始时间 = 0，结束时间 = 6
- 讲座 D：开始时间 = 5，结束时间 = 7

如果我们按照结束时间排序，得到的顺序是：A, B, C, D。

(1) 选择讲座 A（结束时间 4），此时最后选择的结束时间为 4。

(2) 不能选择讲座 B（开始时间 3 < 结束时间 4）。

(3) 不能选择讲座 C（开始时间 0 < 结束时间 4）。

(4) 选择讲座 D（开始时间 5 $\geq$ 结束时间 4），此时最后选择的结束时间为 7。

最终选择的讲座为 A 和 D，共 2 个讲座。

然而，如果我们选择讲座 B 而不是 A：

(1) 选择讲座 B（结束时间 5），此时最后选择的结束时间为 5。

(2) 选择讲座 D（开始时间 5 $\geq$ 结束时间 5），此时最后选择的结束时间为 7。

最终选择的讲座为 B 和 D，共 2 个讲座。

在这种情况下，虽然贪婪算法选择了结束时间最早的讲座，但未能找到最优解。

3.4 习题 3.1.68

a) 试解释为什么一个序列最多只有一个多数元素。

b) 给定序列 2,1,3,3,2,3, 试列出 Boyer-Moore 多数投票算法的步骤。

c) 用伪代码描述 Boyer-Moore 多数投票算法。

d) 试解释你如何确定 Boyer-Moore 算法产生的候选多数元素确实是一个多数元素。

3.4.1 a) 解释为什么一个序列最多只有一个多数元素

一个多数元素是指在序列中出现次数超过一半的元素。如果一个序列有 n 个元素，那么多数元素的出现次数必须大于 $\frac{n}{2}$。假设存在两个不同的多数元素 x 和 y，则 x 和 y 的出现次数加起来至少为 $n+1$，这与每个元素出现次数必须小于等于 n 的条件相矛盾。因此，一个序列最多只有一个多数元素。

3.4.2 b) 列出 Boyer-Moore 多数投票算法的步骤

给定序列：2, 1, 3, 3, 2, 3。

(1) 初始化：候选元素 ‘candidate‘ = None，计数器 ‘count‘ = 0。

(2) 遍历序列：

- 读取元素 2：
  - $count = 0 \rightarrow candidate = 2$，$count = 1$。
- 读取元素 1：
  - $count = 1 \rightarrow count = 0$。
- 读取元素 3：
  - $count = 0 \rightarrow candidate = 3$，$count = 1$。
- 读取元素 3：
  - $count = 1 \rightarrow count = 2$。
- 读取元素 2：
  - $count = 2 \rightarrow count = 1$。
- 读取元素 3：
  - $count = 1 \rightarrow count = 2$。

最终候选元素为 3。

3.4.3 c) 用伪代码描述 Boyer-Moore 多数投票算法

```
function BoyerMooreMajorityVote(nums):
    candidate = None
    count = 0

    for num in nums:
        if count == 0:
            candidate = num
            count = 1
        else if num == candidate:
            count += 1
        else:
            count -= 1

    return candidate
```

3.4.4 d) 解释如何确定 Boyer-Moore 算法产生的候选多数元素确实是一个多数元素

Boyer-Moore 算法的候选元素在算法结束后被称为候选多数元素。要验证该候选元素是否为真正的多数元素，可以再次遍历整个序列，计算候选元素的出现次数。如果该元素的出现次数大于 $\frac{n}{2}$，则它确实是多数元素。这个额外的验证步骤确保了算法的准确性，因为在某些情况下，候选元素可能并不是多数元素。

3.5 习题 3.2.8

8. 对下列每个函数求最小的整数 n 使得 $f(x)$ 是 $O(x^n)$ 的。

a) $f(x) = 2x^2 + x^3 \log x$

b) $f(x) = 3x^5 + (\log x)^4$

c) $f(x) = \frac{x^4+x^2+1}{x^4+1}$

d) $f(x) = \frac{x^3+5\log x}{x^4+1}$

证明. a) $f(x) = 2x^2 + x^3 \log x$

- 分析：$2x^2$ 是 $O(x^2)$，$x^3 \log x$ 是 $O(x^3 \log x)$
- 由于 $x^3 \log x$ 增长得比 x^2 快，因此 $f(x)$ 的主导项是 $x^3 \log x$
- 结论：最小的整数 n 是 3，使得 $f(x)$ 是 $O(x^3)$

b) $f(x) = 3x^5 + (\log x)^4$

- 分析：$3x^5$ 是 $O(x^5)$，$(\log x)^4$ 是 $O((\log x)^4)$
- 由于 x^5 增长得比 $(\log x)^4$ 快，因此 $f(x)$ 的主导项是 $3x^5$
- 结论：最小的整数 n 是 5，使得 $f(x)$ 是 $O(x^5)$

c) $f(x) = \frac{x^4+x^2+1}{x^4+1}$

- 分析：分子和分母的最高次项都是 x^4
- 当 x 很大时，$f(x) \approx \frac{x^4}{x^4} = 1$
- 结论：最小的整数 n 是 0，使得 $f(x)$ 是 $O(1)$

d) $f(x) = \frac{x^3+5\log x}{x^4+1}$

- 分析：分母的最高次项是 x^4，分子的最高次项是 x^3
- 当 x 很大时，$f(x) \approx \frac{x^3}{x^4} = \frac{1}{x}$
- 结论：最小的整数 n 是 -1，使得 $f(x)$ 是 $O(x^{-1})$

□

3.6 习题 3.2.22

将函数 10^0、n^1、$\log^3 n$、$\sqrt{n}\log n$、$n^2 + n$ 和 $(n!)^2$ 排成一列使得每个函数是大 O 下一个函数。

- 10^0
- n^1
- $\log^3 n$
- $\sqrt{n}\log n$
- $n^2 + n$
- $(n!)^2$

3.7 习题 3.2.42

假定 $f(x)$ 是 $O(g(x))$ 的。能否推断出 $2^{f(x)}$ 是 $O(2^{g(x)})$ 的?

证明. 假设 $f(x)$ 是 $O(g(x))$ 的，这意味着存在常数 $C > 0$ 和 x_0，使得对于所有 $x \geq x_0$，都有：

$$|f(x)| \leq C \cdot |g(x)|$$

我们需要证明 $2^{f(x)}$ 是 $O(2^{g(x)})$ 的，即存在常数 $K > 0$ 和 x_1，使得对于所有 $x \geq x_1$，都有：

$$2^{f(x)} \leq K \cdot 2^{g(x)}$$

由于 $f(x) \leq C \cdot g(x)$，我们可以取指数：

$$2^{f(x)} \leq 2^{C \cdot g(x)}$$

令 $K = 2^C$，则有：

$$2^{f(x)} \leq K \cdot 2^{g(x)}$$

因此，存在常数 $K = 2^C$ 使得对于所有 $x \geq x_0$，都有 $2^{f(x)} \leq K \cdot 2^{g(x)}$。这表明 $2^{f(x)}$ 是 $O(2^{g(x)})$ 的。
综上所述，如果 $f(x)$ 是 $O(g(x))$ 的，则 $2^{f(x)}$ 是 $O(2^{g(x)})$ 的。 □

3.8 习题 3.2.56

证明 $\lfloor xy \rfloor$ 是 $\Omega(xy)$。

证明. 我们需要证明存在常数 $c > 0$ 和 x_0, y_0，使得对于所有 $x \geq x_0$ 和 $y \geq y_0$，都有：

$$\lfloor xy \rfloor \geq c \cdot xy$$

由于 $\lfloor xy \rfloor$ 是 xy 的下取整，因此有：

$$\lfloor xy \rfloor \leq xy < \lfloor xy \rfloor + 1$$

这意味着：

$$\lfloor xy \rfloor \geq xy - 1$$

选择 $c = 1$，则对于所有 $x \geq 1$ 和 $y \geq 1$，都有：

$$\lfloor xy \rfloor \geq xy - 1 \geq \frac{1}{2} \cdot xy$$

因此，存在常数 $c = \frac{1}{2}$ 和 $x_0 = 1$，$y_0 = 1$，使得对于所有 $x \geq x_0$ 和 $y \geq y_0$，都有：

$$\lfloor xy \rfloor \geq c \cdot xy$$

综上所述，$\lfloor xy \rfloor$ 是 $\Omega(xy)$。 □

3.9 习题 3.3.12

12. 考虑下面的算法，以 n 个整数 $a_1, a_2, \ldots, a_n$ 的序列作为输入，生成一个矩阵 $M = (m_{ij})$ 作为输出，其中对于 $j \geq i$ 时 m_{ij} 是整数序列 $a_i, a_{i+1}, \ldots, a_j$ 中的最小项，否则 $m_{ij} = 0$。初始化 M 使得当 $j \geq i$ 时 $m_{ij} = a_i$ 否则 $m_{ij} = 0$

for $i := 1$ to n

for $j := i + 1$ to n

for $k := i + 1$ to j

$m_{ij} := \min(m_{ij}, a_k)$

return $M = (m_{ij})$ 其中m_{ij} 是$a_i, a_{i+1}, \ldots, a_j$ 中的最小项

a) 证明这个算法使用 $O(n^3)$ 次比较来计算矩阵 M。

b) 证明这个算法使用 $\Omega(n^3)$ 次比较来计算矩阵 M。利用该事实以及 a) 得出结论该算法使用 $\Theta(n^3)$ 次比较。[提示：在算法的两层外循环中只考虑当 $i \leq n/4$ 和 $j \geq 3n/4$ 的情形。]

证明. a) 证明这个算法使用 $O(n^3)$ 次比较来计算矩阵 M。

我们来分析算法的时间复杂度。算法有三个嵌套的循环：

- 外层循环从 1 到 n，执行 n 次。
- 中层循环从 $i+1$ 到 n，在最坏情况下执行 $n-i$ 次。
- 内层循环从 $i+1$ 到 j，在最坏情况下执行 $j-i$ 次。

因此，总的比较次数为：

$$\sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{k=i+1}^{j} 1$$

我们可以将其简化为：

$$\sum_{i=1}^{n} \sum_{j=i+1}^{n} (j-i)$$

进一步简化为：

$$\sum_{i=1}^{n} \sum_{j=i+1}^{n} j - \sum_{i=1}^{n} \sum_{j=i+1}^{n} i$$

这两个和分别为：

$$\sum_{i=1}^{n} \left(\frac{(n-i)(n-i+1)}{2} \right) - \sum_{i=1}^{n} (n-i)i$$

这两个和的复杂度都是 $O(n^3)$，因此总的比较次数为 $O(n^3)$。

b) 证明这个算法使用 $\Omega(n^3)$ 次比较来计算矩阵 M。

我们只考虑当 $i \leq n/4$ 和 $j \geq 3n/4$ 的情形。在这种情况下，内层循环的范围为 $j-i$，即至少为 $n/2$。因此，总的比较次数至少为：

$$\sum_{i=1}^{n/4} \sum_{j=3n/4}^{n} (j-i)$$

这两个和分别为：

$$\sum_{i=1}^{n/4} \left(\frac{(n/4)(n/4+1)}{2} \right) - \sum_{i=1}^{n/4} (n/4)i$$

这两个和的复杂度都是 $\Omega(n^3)$，因此总的比较次数为 $\Omega(n^3)$。

综上所述，算法的时间复杂度为 $\Theta(n^3)$。

□

3.10 习题 3.3.20

当你将问题的输入规模从 n 翻倍到 $2n$ 时对解题所需时间有什么影响？假定算法解决输入规模为 n 的问题时所需的毫秒数为如下函数。[尽可能将答案表达得简单些，或者一个比值或者一个差值。你的答案可以是 n 的函数或常量。]

a) $\log\log n$

b) $\log n$

c) $100n$

d) $n\log n$

e) n^2

f) n^3

g) 2^n

证明. a) $\log\log n$：

$$\frac{\log\log(2n)}{\log\log n} = \frac{\log(\log 2 + \log n)}{\log\log n} \approx \frac{\log\log n}{\log\log n} = 1$$

因此，时间复杂度几乎不变。

b) $\log n$：

$$\frac{\log(2n)}{\log n} = \frac{\log 2 + \log n}{\log n} = 1 + \frac{\log 2}{\log n} \approx 1 + \frac{0.3010}{\log n}$$

因此，时间复杂度增加了一个常数。

c) $100n$：

$$\frac{100(2n)}{100n} = 2$$

因此，时间复杂度翻倍。

d) $n\log n$：

$$\frac{(2n)\log(2n)}{n\log n} = 2\cdot\frac{\log(2n)}{\log n} = 2\left(1 + \frac{\log 2}{\log n}\right) \approx 2\left(1 + \frac{0.3010}{\log n}\right)$$

因此，时间复杂度大约增加了两倍。

e) n^2：

$$\frac{(2n)^2}{n^2} = \frac{4n^2}{n^2} = 4$$

因此，时间复杂度增加了四倍。

f) n^3：

$$\frac{(2n)^3}{n^3} = \frac{8n^3}{n^3} = 8$$

因此，时间复杂度增加了八倍。

g) 2^n：

$$\frac{2^{2n}}{2^n} = 2^n$$

因此，时间复杂度呈指数级增长。

□

3.11 习题 3.3.42

找出通过在每一步加入一个和那些已安排讲座兼容的结束时间最早的讲座的方式安排最多讲座的贪婪算法的复杂度（3.1 节算法 7）。假设讲座还没有按最早结束时间排序，并且假设排序的最坏情形时间复杂度是 $O(n \log n)$。

证明. 该贪婪算法的步骤如下：

(1) 将所有讲座按结束时间进行排序，时间复杂度为 $O(n \log n)$。

(2) 初始化一个空的安排列表。

(3) 依次检查每个讲座，如果该讲座与已安排的讲座不冲突，则将其加入安排列表中。这个步骤需要遍历所有讲座，时间复杂度为 $O(n)$。

因此，总的时间复杂度为：

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

综上所述，通过在每一步加入一个和那些已安排讲座兼容的结束时间最早的讲座的方式安排最多讲座的贪婪算法的复杂度是 $O(n \log n)$。 □

3.12 习题 3.3.48

48. 计算乘积 ABC 的最佳次序是什么，如果 A、B 和 C 分别是 3×9、9×4 和 4×2 矩阵？

证明. 我们需要找到计算矩阵乘积 ABC 的最佳次序，以最小化标量乘法的次数。我们可以通过计算不同次序的标量乘法次数来确定最佳次序。

设 A 是 3×9 矩阵，B 是 9×4 矩阵，C 是 4×2 矩阵。

计算次序 $(AB)C$：

- 计算 AB 的标量乘法次数：$3 \times 9 \times 4 = 108$
- 结果是一个 3×4 矩阵
- 计算 $(AB)C$ 的标量乘法次数：$3 \times 4 \times 2 = 24$
- 总标量乘法次数：$108 + 24 = 132$

计算次序 $A(BC)$：

- 计算 BC 的标量乘法次数：$9 \times 4 \times 2 = 72$
- 结果是一个 9×2 矩阵
- 计算 $A(BC)$ 的标量乘法次数：$3 \times 9 \times 2 = 54$
- 总标量乘法次数：$72 + 54 = 126$

比较两种次序的总标量乘法次数：

- $(AB)C$ 的总标量乘法次数：132
- $A(BC)$ 的总标量乘法次数：126

因此，计算乘积 ABC 的最佳次序是 $A(BC)$，总标量乘法次数为 126。 □

Homework 4 第四章作业

4.1 习题 4.1.14

14. 下列各式的商和余数是多少？

a) $44 \div 8$

b) $777 \div 21$

c) $-123 \div 19$

d) $-1 \div 23$

e) $-2002 \div 87$

f) $0 \div 17$

g) $1234567 \div 1001$

h) $-100 \div 101$

Solution 1. *a)* $44 \div 8 = 5$ 余数 4

b) $777 \div 21 = 37$ 余数 0

c) $-123 \div 19 = -7$ 余数 10

d) $-1 \div 23 = -1$ 余数 22

e) $-2002 \div 87 = -24$ 余数 86

f) $0 \div 17 = 0$ 余数 0

g) $1234567 \div 1001 = 1233$ 余数 334

h) $-100 \div 101 = -1$ 余数 1

4.2 习题 4.1.18

18. 假设 a 和 b 是整数，$a \equiv 11 \pmod{19}$ 且 $b \equiv 3 \pmod{19}$。试找出满足 $0 \leq c \leq 18$ 的整数 c 使得

a) $c \equiv 13a \pmod{19}$

b) $c \equiv 8b \pmod{19}$

c) $c \equiv a - b \pmod{19}$

d) $c \equiv 7a + 3b \pmod{19}$

e) $c \equiv 2a^2 + 3b^2 \pmod{19}$

f) $c \equiv a^3 + 4b^3 \pmod{19}$

Solution 2. *a)* $c \equiv 13 \cdot 11 \pmod{19} \equiv 143 \pmod{19} \equiv 10$

b) $c \equiv 8 \cdot 3 \pmod{19} \equiv 24 \pmod{19} \equiv 5$

c) $c \equiv 11 - 3 \pmod{19} \equiv 8$

d) $c \equiv 7 \cdot 11 + 3 \cdot 3 \pmod{19} \equiv 77 + 9 \pmod{19} \equiv 86 \pmod{19} \equiv 10$

e) $c \equiv 2 \cdot 11^2 + 3 \cdot 3^2 \pmod{19} \equiv 2 \cdot 121 + 3 \cdot 9 \pmod{19} \equiv 242 + 27 \pmod{19} \equiv 269 \pmod{19} \equiv 3$

f) $c \equiv 11^3 + 4 \cdot 3^3 \pmod{19} \equiv 1331 + 4 \cdot 27 \pmod{19} \equiv 1331 + 108 \pmod{19} \equiv 1439 \pmod{19} \equiv 15$

4.3 习题 4.1.42

42. 证明：如果 a, b, c 和 m 为整数使得 $m \geq 2, c > 0$，且 $a \equiv b \pmod{m}$，则 $ac \equiv bc \pmod{mc}$。

证明. 由于 $a \equiv b \pmod{m}$，我们有 $a = b + km$，其中 k 是某个整数。

将两边乘以 c，得到：

$$ac = (b + km)c = bc + kmc$$

因此，$ac \equiv bc \pmod{mc}$。 □

4.4 习题 4.1.46

46. 证明：如果 n 是一个奇正整数，则 $n^2 \equiv 1 \pmod{8}$。

证明. 设 n 是一个奇正整数，则 n 可以表示为 $n = 2k + 1$，其中 k 是某个整数。

计算 n^2：

$$n^2 = (2k + 1)^2 = 4k^2 + 4k + 1 = 4k(k + 1) + 1$$

由于 $k(k + 1)$ 是一个偶数，因此 $4k(k + 1)$ 是 8 的倍数。

因此，$n^2 \equiv 1 \pmod{8}$。 □

4.5 习题 4.2.6

6. 把下列整数从二进制表示转换为八进制表示。

a) $(11110111)_2$

b) $(101010101010)_2$

c) $(111011101110111)_2$

d) $(101010101010101)_2$

Solution 3. *a)* $(11110111)_2 = (367)_8$

b) $(101010101010)_2 = (5252)_8$

c) $(111011101110111)_2 = (73567)_8$

d) $(101010101010101)_2 = (52525)_8$

4.6 习题 4.2.24

24. 找出下列每一对数的和与积。答案用十六进制表示。

a) $(1AE)_{16}, (BBC)_{16}$

b) $(20CBA)_{16}, (A01)_{16}$

c) $(ABCDE)_{16}, (1111)_{16}$

d) $(E0000E)_{16}, (BAAA)_{16}$

Solution 4. *a)* 和：$(1AE + BBC)_{16} = (D6A)_{16}$，积：$(1AE \times BBC)_{16} = (D8D8C)_{16}$

b) 和：$(20CBA + A01)_{16} = (2165B)_{16}$，积：$(20CBA \times A01)_{16} = (14A5D3A)_{16}$

c) 和：$(ABCDE + 1111)_{16} = (ACEEF)_{16}$，积：$(ABCDE \times 1111)_{16} = (CBA987EF)_{16}$

d) 和：$(E0000E + BAAA)_{16} = (E0BAA8)_{16}$，积：$(E0000E \times BAAA)_{16} = (AABF5D5C)_{16}$

4.7 习题 4.2.26

26. 用算法 5 求 $11^6 \mod 645$。

Algorithm 1 快速模指数运算

```
procedure MODULAR_EXPONENTIATION(b : 整数, n = (a_{k-1}a_{k-2}...a_1a_0)_2, m : 正整数)
    x := 1
    power := b mod m
    for i := 0 to k − 1 do
        if a_i = 1 then
            x := (x · power) mod m
        end if
        power := (power · power) mod m
    end for
    return x                                   ▷ x 等于 b^n mod m
end procedure
```

现在我们使用该算法计算 $11^6 \mod 645$。

首先，将 6 转换为二进制形式：

$$6_{10} = 110_2$$

依次计算每一步的结果：

第 0 位：

$a_0 = 0$

$x := 1$

$power := (11 \cdot 11) \mod 645 = 121$

第 1 位：

$a_1 = 1$

$x := (1 \cdot 121) \mod 645 = 121$

$power := (121 \cdot 121) \mod 645 = 14641 \mod 645 = 451$

第 2 位：

$a_2 = 1$

$x := (121 \cdot 451) \mod 645 = 54571 \mod 645 = 391$

最终结果为：

$$11^6 \mod 645 = 391$$

4.8 习题 4.2.32

32. 证明一个正整数能被 11 整除当且仅当它的偶数位十进制数字之和与奇数位十进制数字之和的差能被 11 整除。

证明. 设一个正整数 n 的十进制表示为 $d_k d_{k-1} \ldots d_1 d_0$，其中 d_i 是第 i 位的数字。

定义偶数位数字之和为 S_{even}，奇数位数字之和为 S_{odd}。

我们需要证明 n 能被 11 整除当且仅当 $S_{\text{even}} - S_{\text{odd}}$ 能被 11 整除。

由于 10 的幂次 10^i 对 11 的余数是交替的 1 和 -1，因此：

$$n \equiv d_0 - d_1 + d_2 - d_3 + \ldots \pmod{11}$$

这意味着 $n \equiv S_{\text{even}} - S_{\text{odd}} \pmod{11}$。

因此，n 能被 11 整除当且仅当 $S_{\text{even}} - S_{\text{odd}}$ 能被 11 整除。 □

4.9 习题 4.3.4

4. 求下列整数的素因子分解式。

a) 39

b) 81

c) 101

d) 143

e) 289

f) 899

Solution 5. *a)* $39 = 3 \times 13$

b) $81 = 3^4$

c) 101 是素数

d) $143 = 11 \times 13$

e) $289 = 17^2$

f) $899 = 29 \times 31$

4.10 习题 4.3.16

16. 判断下列各组整数是否两两互素？

a) 21, 34, 55

b) 14, -17, 85

c) 25, 41, 49, 64

d) 17, 18, 19, 23

Solution 6. *a) 21* 和 *34* 互素，*21* 和 *55* 互素，*34* 和 *55* 互素。因此，*21, 34, 55* 两两互素。

b) 14 和 *-17* 互素，*14* 和 *85* 不互素（公因数为 *7*），*-17* 和 *85* 互素。因此，*14, -17, 85* 不两两互素。

c) 25 和 *41* 互素，*25* 和 *49* 互素，*25* 和 *64* 互素，*41* 和 *49* 互素，*41* 和 *64* 互素，*49* 和 *64* 互素。因此，*25, 41, 49, 64* 两两互素。

d) 17 和 *18* 互素，*17* 和 *19* 互素，*17* 和 *23* 互素，*18* 和 *19* 互素，*18* 和 *23* 互素，*19* 和 *23* 互素。因此，*17, 18, 19, 23* 两两互素。

4.11 习题 4.3.32

32. 用欧几里得算法求

a) $\gcd(1,5)$

b) $\gcd(100,101)$

c) $\gcd(123,277)$

d) $\gcd(1529,14039)$

e) $\gcd(1529,14038)$

f) $\gcd(11111,111111)$

Solution 7. *a)* $\gcd(1,5)$

$$5 = 5 \cdot 1 + 0$$
$$\gcd(1,5) = 1$$

b) $\gcd(100,101)$

$$101 = 1 \cdot 100 + 1$$
$$100 = 100 \cdot 1 + 0$$
$$\gcd(100,101) = 1$$

c) $\gcd(123,277)$

$$277 = 2 \cdot 123 + 31$$
$$123 = 3 \cdot 31 + 30$$
$$31 = 1 \cdot 30 + 1$$
$$30 = 30 \cdot 1 + 0$$
$$\gcd(123,277) = 1$$

d) $\gcd(1529,14039)$

$$14039 = 9 \cdot 1529 + 1280$$
$$1529 = 1 \cdot 1280 + 249$$
$$1280 = 5 \cdot 249 + 35$$
$$249 = 7 \cdot 35 + 4$$
$$35 = 8 \cdot 4 + 3$$
$$4 = 1 \cdot 3 + 1$$
$$3 = 3 \cdot 1 + 0$$
$$\gcd(1529,14039) = 1$$

e) $\gcd(1529, 14038)$

$$14038 = 9 \cdot 1529 + 77$$
$$1529 = 19 \cdot 77 + 66$$
$$77 = 1 \cdot 66 + 11$$
$$66 = 6 \cdot 11 + 0$$
$$\gcd(1529, 14038) = 11$$

f) $\gcd(11111, 111111)$

$$111111 = 10 \cdot 11111 + 1$$
$$11111 = 11111 \cdot 1 + 0$$
$$\gcd(11111, 111111) = 1$$

4.12 习题 4.3.44

44. 利用扩展欧几里得算法把 $\gcd(1001, 100001)$ 表示成 1001 和 100001 的线性组合。

Solution 8. 使用扩展欧几里得算法，我们得到：

$$100001 = 99 \cdot 1001 + 1000$$
$$1001 = 1 \cdot 1000 + 1$$
$$1000 = 1000 \cdot 1 + 0$$

因此，$\gcd(1001, 100001) = 1$。
反向操作这些步骤，我们得到：

$$1 = 1001 - 1 \cdot 1000$$
$$1000 = 100001 - 99 \cdot 1001$$
$$1 = 1001 - 1 \cdot (100001 - 99 \cdot 1001) = 1001 - 100001 + 99 \cdot 1001 = 100 \cdot 1001 - 100001$$

因此，$\gcd(1001, 100001) = 100 \cdot 1001 - 100001$。

4.13 习题 4.3.50

50. 证明如果 a, b 和 m 为整数使得 $m \geq 2$ 且 $a \equiv b \pmod{m}$，则 $\gcd(a, m) = \gcd(b, m)$。

证明. 由于 $a \equiv b \pmod{m}$，我们有 $a = b + km$，其中 k 是某个整数。
因此，$\gcd(a, m) = \gcd(b + km, m) = \gcd(b, m)$。 □

4.14 习题 4.4.6

6. 用例 2 中的方法对下列每对互素的整数找出 $a \mod m$ 的逆。
a) $a = 2, m = 17$

b) $a = 34, m = 89$

c) $a = 144, m = 233$

d) $a = 200, m = 1001$

6. 用例 2 中的方法对下列每对互素的整数找出 $a \mod m$ 的逆。

a) $a = 2, m = 17$

b) $a = 34, m = 89$

c) $a = 144, m = 233$

d) $a = 200, m = 1001$

例 2：找出 101 模 4620 的逆。解：为了完整性，我们给出用来计算 101 模 4620 的逆的全部步骤。（只有最后一步超出了 4.3 节介绍的方法，并在那里的例 17 中做了解释。）首先，用欧几里得算法证明 gcd(101,4620)=1。然后颠倒步骤找出贝祖系数 a 和 b 使得 $101a + 4620b = 1$。于是可推出 a 是 101 模 4620 的一个逆。欧几里得算法用于寻找 gcd(101,4620) 的步骤是：

$$
\begin{aligned}
4620 &= 45 \cdot 101 + 75 \\
101 &= 1 \cdot 75 + 26 \\
75 &= 2 \cdot 26 + 23 \\
26 &= 1 \cdot 23 + 3 \\
23 &= 7 \cdot 3 + 2 \\
3 &= 1 \cdot 2 + 1 \\
2 &= 2 \cdot 1
\end{aligned}
$$

因为最后非零余数是 1，所以可知 gcd(101,4620)=1。可以通过反向操作这些步骤，用连续的余数对表示 gcd(101,4620)=1，从而找出 101 和 4620 的贝祖系数，在每一步通过将余数表示成除数和被除数的线性组合来消除余数。我们得到：

$$
\begin{aligned}
1 &= 3 - 1 \cdot 2 \\
&= 3 - 1 \cdot (23 - 7 \cdot 3) = -1 \cdot 23 + 8 \cdot 3 \\
&= -1 \cdot 23 + 8 \cdot (26 - 1 \cdot 23) = 8 \cdot 26 - 9 \cdot 23 \\
&= 8 \cdot 26 - 9 \cdot (75 - 2 \cdot 26) = -9 \cdot 75 + 26 \cdot 26 \\
&= -9 \cdot 75 + 26 \cdot (101 - 1 \cdot 75) = 26 \cdot 101 - 35 \cdot 75 \\
&= 26 \cdot 101 - 35 \cdot (4620 - 45 \cdot 101) = -35 \cdot 4620 + 1601 \cdot 101
\end{aligned}
$$

$-35 \cdot 4620 + 1601 \cdot 101 = 1$ 告诉我们 -35 和 1601 是 4620 和 101 的贝祖系数，而 1601 是 101 模 4620 的逆。

一旦有了 $a \mod m$ 的逆 a^{-1}，就可以通过在线性同余方程两边同时乘以 a^{-1} 来求解同余方程 $ax = b \mod m$，如例 3 所示。

Solution 9. *a)* 使用扩展欧几里得算法，我们得到 $2 \mod 17$ 的逆是 *9*。

$$\begin{aligned} 17 &= 8 \cdot 2 + 1 \\ 2 &= 2 \cdot 1 + 0 \\ \gcd(2, 17) &= 1 \end{aligned}$$

$$\begin{aligned} 1 &= 17 - 8 \cdot 2 \\ &= 1 \cdot 17 - 8 \cdot 2 \end{aligned}$$

因此，*2* 的逆是 *9*。

b) 使用扩展欧几里得算法，我们得到 $34 \mod 89$ 的逆是 *55*。

$$\begin{aligned} 89 &= 2 \cdot 34 + 21 \\ 34 &= 1 \cdot 21 + 13 \\ 21 &= 1 \cdot 13 + 8 \\ 13 &= 1 \cdot 8 + 5 \\ 8 &= 1 \cdot 5 + 3 \\ 5 &= 1 \cdot 3 + 2 \\ 3 &= 1 \cdot 2 + 1 \\ 2 &= 2 \cdot 1 + 0 \\ \gcd(34, 89) &= 1 \end{aligned}$$

$$\begin{aligned} 1 &= 3 - 1 \cdot 2 \\ &= 3 - 1 \cdot (5 - 1 \cdot 3) = 2 \cdot 3 - 1 \cdot 5 \\ &= 2 \cdot (8 - 1 \cdot 5) - 1 \cdot 5 = 2 \cdot 8 - 3 \cdot 5 \\ &= 2 \cdot 8 - 3 \cdot (13 - 1 \cdot 8) = 5 \cdot 8 - 3 \cdot 13 \\ &= 5 \cdot (21 - 1 \cdot 13) - 3 \cdot 13 = 5 \cdot 21 - 8 \cdot 13 \\ &= 5 \cdot 21 - 8 \cdot (34 - 1 \cdot 21) = 13 \cdot 21 - 8 \cdot 34 \\ &= 13 \cdot (89 - 2 \cdot 34) - 8 \cdot 34 = 13 \cdot 89 - 34 \cdot 55 \end{aligned}$$

因此，*34* 的逆是 *55*。

c) 使用扩展欧几里得算法，我们得到 144 mod 233 的逆是 *89*。

$$
\begin{aligned}
233 &= 1 \cdot 144 + 89 \\
144 &= 1 \cdot 89 + 55 \\
89 &= 1 \cdot 55 + 34 \\
55 &= 1 \cdot 34 + 21 \\
34 &= 1 \cdot 21 + 13 \\
21 &= 1 \cdot 13 + 8 \\
13 &= 1 \cdot 8 + 5 \\
8 &= 1 \cdot 5 + 3 \\
5 &= 1 \cdot 3 + 2 \\
3 &= 1 \cdot 2 + 1 \\
2 &= 2 \cdot 1 + 0 \\
\gcd(144, 233) &= 1
\end{aligned}
$$

$$
\begin{aligned}
1 &= 3 - 1 \cdot 2 \\
&= 3 - 1 \cdot (5 - 1 \cdot 3) = 2 \cdot 3 - 1 \cdot 5 \\
&= 2 \cdot (8 - 1 \cdot 5) - 1 \cdot 5 = 2 \cdot 8 - 3 \cdot 5 \\
&= 2 \cdot 8 - 3 \cdot (13 - 1 \cdot 8) = 5 \cdot 8 - 3 \cdot 13 \\
&= 5 \cdot (21 - 1 \cdot 13) - 3 \cdot 13 = 5 \cdot 21 - 8 \cdot 13 \\
&= 5 \cdot 21 - 8 \cdot (34 - 1 \cdot 21) = 13 \cdot 21 - 8 \cdot 34 \\
&= 13 \cdot (55 - 1 \cdot 34) - 8 \cdot 34 = 13 \cdot 55 - 21 \cdot 34 \\
&= 13 \cdot 55 - 21 \cdot (89 - 1 \cdot 55) = 34 \cdot 55 - 21 \cdot 89 \\
&= 34 \cdot (144 - 1 \cdot 89) - 21 \cdot 89 = 34 \cdot 144 - 55 \cdot 89 \\
&= 34 \cdot 144 - 55 \cdot (233 - 1 \cdot 144) = 89 \cdot 144 - 55 \cdot 233
\end{aligned}
$$

因此，*144* 的逆是 *89*。

d) 使用扩展欧几里得算法，我们得到 200 mod 1001 的逆是 *801*。

$$
\begin{aligned}
1001 &= 5 \cdot 200 + 1 \\
200 &= 200 \cdot 1 + 0 \\
\gcd(200, 1001) &= 1
\end{aligned}
$$

$$
\begin{aligned}
1 &= 1001 - 5 \cdot 200 \\
&= 1 \cdot 1001 - 5 \cdot 200
\end{aligned}
$$

因此，*200* 的逆是 *801*。

4.15 习题 4.4.10

10. 解同余方程 $2x \equiv 7 \pmod{17}$，利用练习 6a 中找到的 2 mod 17 的逆。

Solution 10. 在练习 *6a* 中，我们找到 2 mod 17 的逆是 *9*。我们可以通过将同余方程两边同时乘以 *9* 来求解：

$$2x \equiv 7 \pmod{17}$$

乘以 *9*：

$$9 \cdot 2x \equiv 9 \cdot 7 \pmod{17}$$

简化：

$$18x \equiv 63 \pmod{17}$$

由于 $18 \equiv 1 \pmod{17}$，我们有：

$$x \equiv 63 \pmod{17}$$

计算 63 mod 17：

$$63 \div 17 = 3 \text{ 余} 12$$

因此，解为：

$$x \equiv 12 \pmod{17}$$

4.16 习题 4.4.20

20. 利用中国剩余定理证明中的构造法找出同余方程组 $x \equiv 2 \pmod{3}$，$x \equiv 1 \pmod{4}$ 和 $x \equiv 3 \pmod{5}$ 的所有解。

Solution 11. 根据中国剩余定理，我们可以构造解：

$$x = a_1 M_1 y_1 + a_2 M_2 y_2 + a_3 M_3 y_3$$

其中 $M = 3 \cdot 4 \cdot 5 = 60$，$M_1 = \frac{M}{3} = 20$，$M_2 = \frac{M}{4} = 15$，$M_3 = \frac{M}{5} = 12$。

我们需要找到 y_1, y_2, y_3 使得：

$$20y_1 \equiv 1 \pmod{3}$$
$$15y_2 \equiv 1 \pmod{4}$$
$$12y_3 \equiv 1 \pmod{5}$$

通过计算，我们得到：

$$y_1 \equiv 2 \pmod{3}$$
$$y_2 \equiv 3 \pmod{4}$$
$$y_3 \equiv 3 \pmod{5}$$

因此，解为：

$$x = 2 \cdot 20 \cdot 2 + 1 \cdot 15 \cdot 3 + 3 \cdot 12 \cdot 3 \pmod{60}$$

计算得：

$$x = 80 + 45 + 108 \equiv 233 \pmod{60} \equiv 53 \pmod{60}$$

所以，所有解为：

$$x \equiv 53 \pmod{60}$$

4.17 习题 4.4.23

23. 用反向替换方法求解练习 20 的同余方程组。

Solution 12. 我们从最后一个同余方程开始：

$$x \equiv 3 \pmod 5$$

设 $x = 5k + 3$，代入第二个同余方程：

$$5k + 3 \equiv 1 \pmod 4$$

简化：

$$5k \equiv -2 \pmod 4 \equiv 2 \pmod 4$$

由于 $5 \equiv 1 \pmod 4$，我们有：

$$k \equiv 2 \pmod 4$$

设 $k = 4m + 2$，代入第一个同余方程：

$$x = 5(4m + 2) + 3 = 20m + 13$$

代入第一个同余方程：

$$20m + 13 \equiv 2 \pmod 3$$

简化：

$$20m \equiv -11 \pmod 3 \equiv 1 \pmod 3$$

由于 $20 \equiv 2 \pmod 3$，我们有：

$$2m \equiv 1 \pmod 3$$

乘以 *2* 的逆元：

$$m \equiv 2 \pmod 3$$

设 $m = 3n + 2$，代入 x：

$$x = 20(3n + 2) + 13 = 60n + 53$$

所以，所有解为：

$$x \equiv 53 \pmod{60}$$

4.18 习题 4.4.38

38.

a) 利用费马小定理计算 $30^2 \mod 5$，$3^{302} \mod 7$ 和 $3^{302} \mod 11$。

b) 利用 a 中结果及中国剩余定理计算 $30^{302} \mod 385$。（注意 $385 = 5 \cdot 7 \cdot 11$）

Solution 13. *a)* 利用费马小定理：

$$30^2 \equiv 0 \pmod 5$$
$$3^{302} \equiv 3^{302 \mod 6} \equiv 3^2 \equiv 2 \pmod 7$$
$$3^{302} \equiv 3^{302 \mod 10} \equiv 3^2 \equiv 9 \pmod{11}$$

b) 利用中国剩余定理：

$$x \equiv 0 \pmod 5$$
$$x \equiv 2 \pmod 7$$
$$x \equiv 9 \pmod{11}$$

设 $x = 5k$，代入第二个同余方程：

$$5k \equiv 2 \pmod 7 \implies k \equiv 3 \pmod 7 \implies k = 7m + 3$$

代入 x：

$$x = 5(7m + 3) = 35m + 15$$

代入第三个同余方程：

$$35m+15 \equiv 9 \pmod{11} \implies 2m+4 \equiv 9 \pmod{11} \implies 2m \equiv 5 \pmod{11} \implies m \equiv 8 \pmod{11}$$

设 $m = 11n + 8$，代入 x：

$$x = 35(11n + 8) + 15 = 385n + 295$$

所以，所有解为：

$$x \equiv 295 \pmod{385}$$

4.19 习题 4.5.6

6. 用线性同余生成器 $x_{n+1} = (4x_n + 1) \mod 7$ 和种子 $x_0 = 3$ 生成的伪随机数序列是什么？

Solution 14. 计算伪随机数序列：

$$x_1 = (4 \cdot 3 + 1) \mod 7 = 13 \mod 7 = 6$$
$$x_2 = (4 \cdot 6 + 1) \mod 7 = 25 \mod 7 = 4$$
$$x_3 = (4 \cdot 4 + 1) \mod 7 = 17 \mod 7 = 3$$
$$x_4 = (4 \cdot 3 + 1) \mod 7 = 13 \mod 7 = 6$$

重复

所以，伪随机数序列为：

$$3, 6, 4, 3, 6, 4, \ldots$$

4.20 习题 4.5.34

34. 一个 ISSN 的校验码是否能检测出 ISSN 中每个单错？用证明或反例来解释你的答案。

Solution 15. *ISSN* 的校验码是通过加权和计算的，权重从 *8* 到 *1*。校验码公式为：

$$\sum_{i=1}^{8} i \cdot d_i \equiv 0 \pmod{11}$$

假设有一个单错，即某一位数字 d_i 变为 d_i'，则校验和变为：

$$\sum_{i=1}^{8} i \cdot d_i' \equiv 0 \pmod{11}$$

由于 $d_i \neq d_i'$，因此 $i \cdot (d_i - d_i') \equiv 0 \pmod{11}$，这意味着 $i \cdot k \equiv 0 \pmod{11}$，其中 $k = d_i - d_i'$。

由于 i 的取值范围是 *1* 到 *8*，且 *11* 是质数，因此 $i \cdot k$ 不可能为 *11* 的倍数。因此，*ISSN* 的校验码可以检测出每个单错。

综上所述，*ISSN* 的校验码能检测出 *ISSN* 中每个单错。

Homework 5 第五章习题

5.1 习题 5.1.4

4. 设 $P(n)$ 是命题：对正整数 n 而言，$1^3+2^3+\cdots+n^3=\left(\frac{n(n+1)}{2}\right)^2$。

(1) 命题 $P(1)$ 是什么?

(2) 证明 $P(1)$ 为真，完成基础步骤的证明。

(3) 归纳假设是什么?

(4) 在归纳步骤中你需要证明什么?

(5) 完成归纳步骤。

(6) 解释为什么只要 n 是一个正整数，则上述步骤就可以证明公式为真。

解答:

(1) 命题 $P(1)$ 是：$1^3=\left(\frac{1(1+1)}{2}\right)^2$。

(2) 证明 $P(1)$ 为真：

$$1^3=1=\left(\frac{1\cdot 2}{2}\right)^2=1$$

因此，$P(1)$ 为真。

(3) 归纳假设是：假设对于某个正整数 k，命题 $P(k)$ 为真，即 $1^3+2^3+\cdots+k^3=\left(\frac{k(k+1)}{2}\right)^2$。

(4) 在归纳步骤中需要证明：$P(k+1)$ 为真，即 $1^3+2^3+\cdots+k^3+(k+1)^3=\left(\frac{(k+1)(k+2)}{2}\right)^2$。

(5) 完成归纳步骤：

$$\begin{aligned}1^3+2^3+\cdots+k^3+(k+1)^3&=\left(\frac{k(k+1)}{2}\right)^2+(k+1)^3\\&=\left(\frac{k(k+1)}{2}\right)^2+(k+1)^3=(k+1)^2\left(\frac{k^2}{4}+k+1\right)\\&=(k+1)^2\left(\frac{k^2+4k+4}{4}\right)=(k+1)^2\left(\frac{(k+2)^2}{4}\right)\\&=\left(\frac{(k+1)(k+2)}{2}\right)^2\end{aligned}$$

因此，$P(k+1)$ 为真。

(6) 解释：通过数学归纳法，我们已经证明了 $P(1)$ 为真，并且如果 $P(k)$ 为真，则 $P(k+1)$ 也为真。因此，对于所有正整数 n，命题 $P(n)$ 都为真。

5.2 习题 5.1.14

14. 证明：对所有正整数 n 而言，都有 $\sum_{k=1}^{n}k2^k=(n-1)2^{n+1}+2$。

解答:

(1) 基础步骤：当 $n=1$ 时，

$$\sum_{k=1}^{1}k2^k=1\cdot 2^1=2$$

$$(1-1)2^{1+1}+2=0\cdot 4+2=2$$

因此，命题对 $n=1$ 成立。

(2) 归纳假设：假设对于某个正整数 k，命题成立，即

$$\sum_{k=1}^{k} k2^k = (k-1)2^{k+1} + 2$$

(3) 归纳步骤：证明对于 $k+1$，命题也成立。

$$\sum_{k=1}^{k+1} k2^k = \sum_{k=1}^{k} k2^k + (k+1)2^{k+1}$$

根据归纳假设，

$$= (k-1)2^{k+1} + 2 + (k+1)2^{k+1}$$

$$= (k-1+k+1)2^{k+1} + 2 = (2k)2^{k+1} + 2 = (k+1-1)2^{(k+1)+1} + 2$$

因此，命题对 $k+1$ 也成立。

(4) 结论：通过数学归纳法，我们证明了命题对所有正整数 n 成立。

5.3 习题 5.1.32

32. 证明：只要 n 是一个正整数，则 $n^2 + 2n$ 可被 3 整除。

解答:

(1) 基础步骤：当 $n = 1$ 时，

$$1^2 + 2 \cdot 1 = 1 + 2 = 3$$

3 可被 3 整除，因此命题对 $n = 1$ 成立。

(2) 归纳假设：假设对于某个正整数 k，命题成立，即 $k^2 + 2k$ 可被 3 整除。

(3) 归纳步骤：证明对于 $k+1$，命题也成立。

$$(k+1)^2 + 2(k+1) = k^2 + 2k + 1 + 2 = k^2 + 2k + 3$$

根据归纳假设，$k^2 + 2k$ 可被 3 整除，因此 $k^2 + 2k + 3$ 也可被 3 整除。

(4) 结论：通过数学归纳法，我们证明了命题对所有正整数 n 成立。

5.4 习题 5.1.60

60. 用数学归纳法证明：当 $p_1, p_2, \ldots, p_n$ 都是命题时，则 $\neg(p_1 \vee p_2 \vee \cdots \vee p_n)$ 等价于 $\neg p_1 \wedge \neg p_2 \wedge \neg p_3 \wedge \cdots \wedge \neg p_n$。

解答:

(1) 基础步骤：当 $n = 1$ 时，

$$\neg(p_1) = \neg p_1$$

因此，命题对 $n = 1$ 成立。

(2) 归纳假设：假设对于某个正整数 k，命题成立，即

$$\neg(p_1 \vee p_2 \vee \cdots \vee p_k) = \neg p_1 \wedge \neg p_2 \wedge \cdots \wedge \neg p_k$$

(3) 归纳步骤：证明对于 $k+1$，命题也成立。

$$\neg(p_1 \vee p_2 \vee \cdots \vee p_k \vee p_{k+1}) = \neg((p_1 \vee p_2 \vee \cdots \vee p_k) \vee p_{k+1})$$

根据德摩根定律，

$$= \neg(p_1 \vee p_2 \vee \cdots \vee p_k) \wedge \neg p_{k+1}$$

根据归纳假设，

$$= (\neg p_1 \wedge \neg p_2 \wedge \cdots \wedge \neg p_k) \wedge \neg p_{k+1}$$

$$= \neg p_1 \wedge \neg p_2 \wedge \cdots \wedge \neg p_k \wedge \neg p_{k+1}$$

因此，命题对 $k+1$ 也成立。

(4) 结论：通过数学归纳法，我们证明了命题对所有正整数 n 成立。

5.5 习题 5.2.6

6.

(1) 确定只用 3 分和 10 分的邮票可以构成多少数量的邮资。

(2) 用数学归纳法原理证明你对 a 的回答。注意必须明确陈述归纳步骤中的归纳假设。

(3) 用强归纳法证明你对 a 的回答。在该证明中，归纳假设与用数学归纳法原理证明中的归纳假设有什么不同?

Solution 16. ***(1)*** 只用 *3* 分和 *10* 分的邮票可以构成的邮资是所有大于等于 *8* 的整数。

(2) 用数学归纳法原理证明：

基例：当邮资为 *8* 分时，可以用两个 *3* 分邮票和一个 *10* 分邮票构成，即 $8 = 3 \times 2 + 10 - 10$。

归纳假设：假设对于某个 $k \geq 8$，邮资 k 可以用 *3* 分和 *10* 分的邮票构成。

归纳步骤：考虑邮资 $k+1$。如果 $k+1$ 是 *3* 的倍数，则可以直接用 *3* 分邮票构成。如果 $k+1$ 不是 *3* 的倍数，则 $k+1$ 可以表示为 $k+1 = k+3$，根据归纳假设，k 可以用 *3* 分和 *10* 分的邮票构成，再加上一个 *3* 分邮票即可构成 $k+1$。

因此，根据数学归纳法，所有大于等于 *8* 的邮资都可以用 *3* 分和 *10* 分的邮票构成。

(3) 用强归纳法证明：

基例：当邮资为 *8* 分时，可以用两个 *3* 分邮票和一个 *10* 分邮票构成，即 $8 = 3 \times 2 + 10 - 10$。

归纳假设：假设对于所有 n 满足 $8 \leq n \leq k$，邮资 n 可以用 *3* 分和 *10* 分的邮票构成。

归纳步骤：考虑邮资 $k+1$。根据归纳假设，$k-2$ 可以用 *3* 分和 *10* 分的邮票构成，再加上一个 *3* 分邮票即可构成 $k+1$。

因此，根据强归纳法，所有大于等于 *8* 的邮资都可以用 *3* 分和 *10* 分的邮票构成。

区别：在数学归纳法中，归纳假设只考虑 k 的情况，而在强归纳法中，归纳假设考虑了所有小于等于 k 的情况。

5.6 习题 5.2.12

12. 用强归纳法证明：任意正整数 n 都可以写成 2 的不同幂次之和，即可以写成整数的一个子集 $\{2^0 = 1, 2^1 = 2, 2^2 = 4, \ldots\}$ 的和。[提示：对归纳步骤，分别考虑 $k+1$ 是偶数和奇数时的情况。当 $k+1$ 是偶数时，注意 $(k+1)/2$ 是整数。]

Solution 17. 基例：当 $n = 1$ 时，*1* 可以表示为 2^0。

归纳假设：假设对于所有 $n \leq k$，n 可以表示为 *2* 的不同幂次之和。

归纳步骤：考虑 $k+1$ 的情况。

- 如果 $k+1$ 是偶数，则 $k+1=2m$，根据归纳假设，m 可以表示为 *2* 的不同幂次之和，再乘以 *2* 即可表示 $k+1$。
- 如果 $k+1$ 是奇数，则 $k+1=2m+1$，根据归纳假设，m 可以表示为 *2* 的不同幂次之和，再加上 2^0 即可表示 $k+1$。

因此，根据强归纳法，任意正整数 n 都可以写成 *2* 的不同幂次之和。

5.7 习题 5.2.26

26. 设 $P(n)$ 是命题函数。确定对哪些非负整数 n，命题 $P(n)$ 必为真，如果

(1) $P(0)$ 为真；对所有的非负整数 n，如果 $P(n)$ 为真，那么 $P(n+2)$ 为真。

(2) $P(0)$ 为真；对所有的非负整数 n，如果 $P(n)$ 为真，那么 $P(n+3)$ 为真。

(3) $P(0)$ 和 $P(1)$ 为真；对所有的非负整数 n，如果 $P(n)$ 和 $P(n+1)$ 为真，那么 $P(n+2)$ 为真。

(4) $P(0)$ 为真；对所有的非负整数 n，如果 $P(n)$ 为真，那么 $P(n+2)$ 和 $P(n+3)$ 为真。

Solution 18. ***(1)*** 对于所有的偶数 n，命题 $P(n)$ 必为真。

(2) 对于所有的 $n \equiv 0 \pmod 3$，命题 $P(n)$ 必为真。

(3) 对于所有的非负整数 n，命题 $P(n)$ 必为真。

(4) 对于所有的非负整数 n，命题 $P(n)$ 必为真。

5.8 习题 5.2.34

34. 证明：对所有的正整数 n 和 k，

$$\sum_{j=1}^{n} j(j+1)(j+2)\cdots(j+k-1) = \frac{n(n+1)(n+2)\cdots(n+k)}{k+1}$$

[提示：利用练习 33 中的技巧。]

Solution 19. 我们利用数学归纳法来证明这个命题。

基例：当 $n=1$ 时，

$$\sum_{j=1}^{1} j(j+1)(j+2)\cdots(j+k-1) = 1\cdot 2\cdot 3\cdots k = \frac{1\cdot 2\cdot 3\cdots(k+1)}{k+1}$$

该等式成立。

归纳假设：假设对于某个 n，

$$\sum_{j=1}^{n} j(j+1)(j+2)\cdots(j+k-1) = \frac{n(n+1)(n+2)\cdots(n+k)}{k+1}$$

归纳步骤：考虑 $n+1$ 的情况，

$$\sum_{j=1}^{n+1} j(j+1)(j+2)\cdots(j+k-1) = \sum_{j=1}^{n} j(j+1)(j+2)\cdots(j+k-1) + (n+1)(n+2)\cdots(n+k)$$

根据归纳假设，

$$\sum_{j=1}^{n} j(j+1)(j+2)\cdots(j+k-1) = \frac{n(n+1)(n+2)\cdots(n+k)}{k+1}$$

因此，

$$\sum_{j=1}^{n+1} j(j+1)(j+2)\cdots(j+k-1) = \frac{n(n+1)(n+2)\cdots(n+k)}{k+1} + (n+1)(n+2)\cdots(n+k)$$

提取公因子 $(n+1)(n+2)\cdots(n+k)$，

$$\sum_{j=1}^{n+1} j(j+1)(j+2)\cdots(j+k-1) = (n+1)(n+2)\cdots(n+k)\left(\frac{n}{k+1}+1\right)$$

化简，

$$\sum_{j=1}^{n+1} j(j+1)(j+2)\cdots(j+k-1) = (n+1)(n+2)\cdots(n+k)\left(\frac{n+k+1}{k+1}\right)$$

因此，

$$\sum_{j=1}^{n+1} j(j+1)(j+2)\cdots(j+k-1) = \frac{(n+1)(n+2)\cdots(n+k+1)}{k+1}$$

综上所述，命题对所有的正整数 n 和 k 成立。

5.9 习题 5.3.8

8. 给出序列 $\{a_n\}$ 的递归定义，$n = 1, 2, 3, \ldots$，若

(1) $a_n = 4n - 2$

(2) $a_n = 1 + (-1)^n$

(3) $a_n = n(n+1)$

(4) $a_n = n^2$

Solution 20. ***(1)*** $a_n = 4n - 2$

$$\begin{cases} a_1 = 2 \\ a_n = a_{n-1} + 4, \quad n > 1 \end{cases}$$

(2) $a_n = 1 + (-1)^n$

$$\begin{cases} a_1 = 0 \\ a_n = 2 - a_{n-1}, \quad n > 1 \end{cases}$$

(3) $a_n = n(n+1)$

$$\begin{cases} a_1 = 2 \\ a_n = a_{n-1} + 2n, \quad n > 1 \end{cases}$$

(4) $a_n = n^2$

$$\begin{cases} a_1 = 1 \\ a_n = a_{n-1} + 2n - 1, \quad n > 1 \end{cases}$$

5.10 习题 5.3.18

18. 设

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

证明当 n 是正整数时，有

$$A^n = \begin{bmatrix} f_{n+1} & f_n \\ f_n & f_{n-1} \end{bmatrix}$$

证明. 我们使用数学归纳法来证明这个命题。

基例：当 $n = 1$ 时，

$$A^1 = A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

而

$$\begin{bmatrix} f_2 & f_1 \\ f_1 & f_0 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

该等式成立。

归纳假设：假设对于某个正整数 k，有

$$A^k = \begin{bmatrix} f_{k+1} & f_k \\ f_k & f_{k-1} \end{bmatrix}$$

归纳步骤：考虑 $n = k + 1$ 的情况，

$$A^{k+1} = A^k \cdot A = \begin{bmatrix} f_{k+1} & f_k \\ f_k & f_{k-1} \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

计算得，

$$A^{k+1} = \begin{bmatrix} f_{k+1} + f_k & f_{k+1} \\ f_k + f_{k-1} & f_k \end{bmatrix}$$

根据斐波那契数列的定义，

$$f_{k+2} = f_{k+1} + f_k$$

因此，

$$A^{k+1} = \begin{bmatrix} f_{k+2} & f_{k+1} \\ f_{k+1} & f_k \end{bmatrix}$$

综上所述，命题对所有的正整数 n 成立。 □

5.11 习题 5.3.24

24. 给出下述集合的递归定义：

(1) 正奇数集合

(2) 3 的正整数次幂的集合

(3) 整系数多项式的集合

Solution 21. ***(1)*** 正奇数集合

- 基础步骤：*1* 是正奇数。
- 归纳步骤：如果 n 是正奇数，则 $n + 2$ 也是正奇数。

(2) *3* 的正整数次幂的集合

- 基础步骤：*3* 是 *3* 的正整数次幂。
- 归纳步骤：如果 n 是 *3* 的正整数次幂，则 $3n$ 也是 *3* 的正整数次幂。

(3) 整系数多项式的集合

- 基础步骤：常数多项式 0 和 1 是整系数多项式。
- 归纳步骤：如果 $p(x)$ 和 $q(x)$ 是整系数多项式，则 $p(x)+q(x)$ 和 $p(x)\cdot q(x)$ 也是整系数多项式。

5.12 习题 5.3.26

26. 设 S 是一个正整数集合，定义如下：

- 基础步骤：$1 \in S$。
- 归纳步骤：如果 $n \in S$，则 $3n+2 \in S$ 且 $n^2 \in S$。

(1) 证明如果 $n \in S$，则 $n \equiv 1 \pmod 4$。

(2) 证明存在一个整数 $m \equiv 1 \pmod 4$ 不属于 S。

Solution 22. ***(1)*** 证明如果 $n \in S$，则 $n \equiv 1 \pmod 4$。

我们使用数学归纳法来证明这个命题。

基例：当 $n = 1$ 时，

$$1 \equiv 1 \pmod 4$$

该等式成立。

归纳假设：假设对于某个正整数 k，$k \in S$ 且 $k \equiv 1 \pmod 4$。

归纳步骤：考虑 $3k+2$ 和 k^2 的情况，

$$3k+2 \equiv 3\cdot 1+2 \equiv 5 \equiv 1 \pmod 4$$

$$k^2 \equiv 1^2 \equiv 1 \pmod 4$$

因此，根据数学归纳法，如果 $n \in S$，则 $n \equiv 1 \pmod 4$。

(2) 证明存在一个整数 $m \equiv 1 \pmod 4$ 不属于 S。

假设 $m = 5$，我们证明 $5 \notin S$。

根据集合 S 的定义，S 中的元素只能通过基础步骤和归纳步骤生成。基础步骤中只有 $1 \in S$。归纳步骤中，如果 $n \in S$，则 $3n+2 \in S$ 且 $n^2 \in S$。

我们可以验证，5 不能通过基础步骤和归纳步骤生成：

- $1 \in S$，$3\cdot 1+2 = 5 \in S$，但 5 不能通过 n^2 生成。
- $3\cdot 1+2 = 5 \in S$，但 5 不能通过 n^2 生成。

因此，$5 \notin S$，即存在一个整数 $m \equiv 1 \pmod 4$ 不属于 S。

5.13 习题 5.3.46

46. 用结构归纳法证明：满二叉树 T 的树叶数 $l(T)$ 比 T 的内点数 $i(T)$ 多 1。

证明. 我们使用结构归纳法来证明这个命题。

基例：当 T 是只有一个节点的满二叉树时，

$$l(T) = 1, \quad i(T) = 0$$

因此，

$$l(T) = i(T) + 1$$

该等式成立。

归纳假设：假设对于任意满二叉树 T，如果 T 的树叶数 $l(T)$ 比 T 的内点数 $i(T)$ 多 1，则 T 的子树也满足该性质。

归纳步骤：考虑一个有两个子树 T_1 和 T_2 的满二叉树 T，

$$l(T) = l(T_1) + l(T_2)$$

$$i(T) = i(T_1) + i(T_2) + 1$$

根据归纳假设，

$$l(T_1) = i(T_1) + 1$$

$$l(T_2) = i(T_2) + 1$$

因此，

$$l(T) = (i(T_1) + 1) + (i(T_2) + 1) = i(T_1) + i(T_2) + 2$$

$$i(T) = i(T_1) + i(T_2) + 1$$

所以，

$$l(T) = i(T) + 1$$

综上所述，命题对所有的满二叉树 T 成立。 □

5.14 习题 5.4.32

32. 设计求一个序列的第 n 项的递归算法，该序列定义成：$a_0 = 1, a_1 = 2, a_2 = 3$，而且对 $n = 3, 4, 5, \ldots$ 来说有 $a_n = a_{n-1} + a_{n-2} + a_{n-3}$。

Solution 23. 递归算法如下：

Algorithm 2 递归求序列第 n 项

```
function SEQUENCE(n)
    if n = 0 then
        return 1
    else if n = 1 then
        return 2
    else if n = 2 then
        return 3
    else
        return SEQUENCE(n − 1) + SEQUENCE(n − 2) + SEQUENCE(n − 3)
    end if
end function
```

5.15 习题 5.4.34

34. 求练习 32 的序列的递归算法与迭代算法，哪个算法更有效?

Solution 24. 迭代算法如下：

递归算法的时间复杂度是指数级的，因为每次递归调用会产生三个新的递归调用。而迭代算法的时间复杂度是线性的，因为它只需要一个循环。因此，迭代算法更有效。

Algorithm 3 迭代求序列第 n 项

```
function SEQUENCE(n)
    if n = 0 then
        return 1
    else if n = 1 then
        return 2
    else if n = 2 then
        return 3
    else
        a_0 := 1
        a_1 := 2
        a_2 := 3
        for i := 3 to n do
            a_n := a_2 + a_1 + a_0
            a_0 := a_1
            a_1 := a_2
            a_2 := a_n
        end for
        return a_n
    end if
end function
```

5.16 习题 5.4.44

44. 用归并排序来排序 $4, 3, 2, 5, 1, 8, 7, 6$，说明算法所用的所有步骤。

Solution 25. 归并排序的步骤如下：

初始序列： $[4, 3, 2, 5, 1, 8, 7, 6]$

第一步： 将序列分成两半

$$[4, 3, 2, 5] \quad [1, 8, 7, 6]$$

第二步： 继续分割

$$[4, 3] \quad [2, 5] \quad [1, 8] \quad [7, 6]$$

第三步： 继续分割

$$[4] \quad [3] \quad [2] \quad [5] \quad [1] \quad [8] \quad [7] \quad [6]$$

第四步： 合并排序

$$[3, 4] \quad [2, 5] \quad [1, 8] \quad [6, 7]$$

第五步： 继续合并排序

$$[2, 3, 4, 5] \quad [1, 6, 7, 8]$$

第六步： 最终合并排序

$$[1, 2, 3, 4, 5, 6, 7, 8]$$

最终排序结果为 $[1, 2, 3, 4, 5, 6, 7, 8]$。