Q1.（10+15=25 分）

1. 如下所示，10 个字符对应的出现频率为：

字符	a	b	c	d	e	f	g	h
出现频率	1	1	2	3	5	8	13	21

   请问频率集合的赫夫曼编码是怎样的？

2. 前一问所示的字符出现频率是斐波那契数列。请推广你的结论，当 n 个字符组成的字符集对应的出现次数恰为前 n 个斐波那契数（第 $i(0 \le i \le n-1)$ 个字符的出现次数为 F_i）时，求最优前缀编码。

解：

1.

字符	a	b	c	d	e	f	g	h
编码	1111111	1111110	111110	11110	1110	110	10	0

（答案不唯一）

2. 编码为 $code(c_{n-1}) = 0$，$code(c_{i-1}) = 1code(c_i)$ $(1 \le i \le n-2)$，$code(c_0) = 1^{n-1}$。

   证明：

   引理 1 $\forall k \in \mathcal{N}$，$\sum_{i=0}^{k} F_i = F_{k+2} - 1$。

   引理 2 z 为由哈夫曼算法构建的树 T 的内部节点，则 $z.freq$ 等于以 z 为根的子树上所有叶子结点的频率之和。

   递归定义子树 T_n：

   (a) $T_1.left = c_1, T_1.right = c_0$，$T_1.freq = c_0.freq + c_1.freq = 2$;

   (b) $\forall i,\ 2 \le i \le n-1, T_i.left = c_i, T_i.right = c_{i-1}, T_i.freq = c_i.freq + T_{i-1}.freq$

   由以上性质，可保证树的结构唯一，对应的编码也确定。

Q2.（15 分）假定我们对一个数据结构执行一个由 n 个操作组成的操作序列，当 i 严格为 3 的幂时，第 i 个操作的代价为 $2i$，否则代价为 1。使用聚合分析确定每个操作的摊还代价。

解：

$$\sum_{i=1}^{n} c(i) = \sum_{i=1}^{\lceil \log_3 n \rceil} 2 * 3^i + \sum_{i \neq 3^k} 1$$
$$\leq 3n + n = 4n = O(n)$$
$$\frac{O(n)}{n} = O(1)$$

Q3.（15 分）

假定一个数据文件由 8 位字符组成，其中所有 256 个字符出现的频率大致相同：最高的频率也低于最低频率的 2 倍。证明：在此情况下，赫夫曼编码并不比 8 位固定长度编码更高效。

证：

对于任意两个字符，它们的频率之和大于任何其他字符的频率，因此最初的哈夫曼编码会生成 128 棵具有两个叶子节点的子树。在下一阶段，没有内部节点的频率超过其他节点的两倍以上，因此我们与之前的设置相同。以这种方式继续，哈夫曼编码构建了一棵完全二叉树，高度为 $\lg 256 = 8$，与普通的 8 位长度编码没有更高的效率。**Q4.**（15 分）

证明：编码树的总代价还可以表示为所有内部结点的两个孩子结点的联合频率之和。

证：

递归证明，当叶子数 $n = 1$ 时，结论成立。$n = 2$ 时，设左右儿子为 x 和 y

$$B(T) = f(x)d_T(x) + f(y)d_T(y) = f(x) + f(y)$$

假设当 $n \leq N - 1$ 时结论成立，下证 $n = N$ 时结论成立。

设叶子节点 c_1、c_2 有共同的父亲 p（是兄弟节点），T' 为删除节点 c_1、

c_2 后的树。

$$
\begin{aligned}
B(T) &= \sum_{\text{leaves } l\in T} f(l)d_T(l) \\
&= \sum_{l\neq c_1,c_2} f(l)d_T(l) + f(c_1)(d_T(c_1)-1) + f(c_2)(d_T(c_2)-1) + f(c_1) + f(c_2) \\
&= \sum_{\text{internal nodes } i'\in T'} (f(child_1\ of\ i') + f(child_2\ of\ i')) + f(c_1) + f(c_2) \\
&= \sum_{\text{internal nodes } i\in T} (f(child_1\ of\ i) + f(child_2\ of\ i))
\end{aligned}
$$

Q5.（10+10+10=30 分）

考虑用最少的硬币找 n 分零钱的问题（$n \in [1,99]$）。假定每种硬币的面额都是整数。

1. 设计贪心算法求解找零问题，假定有 5 角、2 角、1 角、5 分、2 分、1 分 6 种面额的硬币。证明你的算法能找到最优解；
2. 假定硬币面额是 c 的幂，即面额为 c^0，c^1，...，c^k，c 和 k 为整数，$c>1$，$k\geq 1$。证明：贪心算法总能找到最优解。
3. 设计一组硬币面额，使得贪心算法不能得到最优解。这组硬币面额种应该包含 1 分，使得对每个 n 都存在找零方案。（给出不能得到最优解的例子）

解：

1. 总是尽可能先选择面额较大找零，直至待找零金额为 0。
2. 给定最优解 $P=(x_0,x_1,\ldots,x_k)$，其中 x_i 表示面值为 c_i 的数量。显然 $x_i<c$ $(\forall i<k)$，否则可以用 1 个 c_{i+1} 替换 c 个 c_i，则解 P 不是最优解。而该解恰为前述贪心算法的输出结果。而该解是唯一满足该性质的解，因为该解恰是 n 在以 c 为基数的表示。
3. 面额为 $\{1,3,4\}$，最优解为 $(3,3)$，贪婪解为 $(1,1,4)$。