参考解答

Sol1.（10 分）与书上章节 11.8 的内容类似。在书中，b 设置为 $\frac{\epsilon}{2n}\max_i v_i$。在这里，令 $b=\frac{\epsilon W}{n}$，使得 $\tilde{w_i}=\lceil\frac{w_i}{b}\rceil b, \hat{w}_i=\lceil\frac{w_i}{b}\rceil$。此外，我们也知道对 $\tilde{w_i}$ 与 $\hat{w}_i$，它们有同样的最优解集合。

首先会排除所有重量 $w_i > W$ 的物品。然后使用定理 6.12 中的算法去求解。最终得到的解的价值至少为 V。故对于原问题，我们找到了价值至少为 V 的解。与书中（11.34）（11.38）类似的，最终的重量至多为 $W+n\cdot(\frac{\epsilon W}{n})=W(1+\epsilon)$。

Sol2.（30 分）若 $A[i]\neq A[j], j>i$，则若 $j=i+1$，我们边找到了所需的索引。否则，我们选择 i,j 中间的索引 k。若 $A[i]\neq A[k]$，则在 i,k 之间继续递归。若 $A[k]\neq A[j]$，则在 k,j 之间继续递归。注意到上述两个条件至少一个会是对的。否则 $A[i]=A[k]=A[j]$，违背了最开始的假设 $A[i]\neq A[j]$。

简单的伪代码如下：

breakpoint(A, i, j) ($i<j$, $A[i]\neq A[j]$)

    If $j=i+1$

        Return i

    Let $k=\lfloor(i+j)/2\rfloor$

    If $A[i]\neq A[k]$

        Return breakpoint(A, i, k)

    If $A[k]\neq A[j]$

        Return breakpoint(A, j, k)

每次递归会对 $|i-j|$ 除以 2，每次只需常数时间的开销判断是否不等。令 $T(n)$ 为运行时间（$|i-j|=O(n)$），则有：

$$T(n)=T(n/2)+O(1),$$

故由主定理，$T(n)=O(\log n)$。

Sol3.（30 分）将向量 v 重写为 $v=(v_1,v_2)$，其中 $|v_1|=|v_2|=|v|/2$。那么：

$$\bar{H}_n \cdot v = \begin{bmatrix} \bar{H}_{n-1}v_1 + \bar{H}_{n-1}v_2 \\ \bar{H}_{n-1}v_1 - \bar{H}_{n-1}v_2 \end{bmatrix}$$

故只需递归地计算 $\bar{H}_{n-1}v_1$ 与 $\bar{H}_{n-1}v_2$，通过对它们做加（减）运算，结合起来便可以得到 $\bar{H}_n v$ 的值。若 $|v|=1$，就会直接返回 $\bar{H}_0 \cdot v$，也就是 v。

简单的伪代码如下：

Compute($\bar{H}_n v$**)**:
  If $|v|=1$
    Return v
  Split $v=(v_1,v_2)$ **where** $|v_1|=|v_2|=|v|/2$
  $x_1 \leftarrow$ **Compute(**$\bar{H}_{n-1}v_1$**)**
  $x_2 \leftarrow$ **Compute(**$\bar{H}_{n-1}v_2$**)**
  Return $v \leftarrow (x_1+x_2, x_1-x_2)$

不妨令 $N=2^n$。算法中，我们每次递归会将问题划分为两个同等规模的子问题，且对 x_1 与 x_2 的运算需要 $O(N)$。故递归关系为：

$$T(N)=2T(N/2)+O(N).$$

利用主定理（$a=2, b=2, d=1$），知时间复杂度为 $O(N\log N)$。

Sol4.（30 分）(a) 假设 w 为当前的总权重。使用该策略每判断错误一次，说明至少有一半的同学会判断错。那么这之后的总权重至多为

$$\frac{1}{4}w+\frac{1}{2}w=\frac{3}{4}w.$$

若在询问 l 次问题后，该策略判断错误 k 次，则可以得到当前总权重 w 的上界：（初始总权重为 n）

$$w \le \left(\frac{3}{4}\right)^k (1)^{l-k} n = \left(\frac{3}{4}\right)^k n.$$

此外，最可靠的同学如果犯了 m 次错误，其权重会为 $\frac{1}{2^m}$。无疑 $\frac{1}{2^m} \le w$。

两相结合便可得到 $k \le \frac{1}{2-\log_2 3}(m+\log_2 n)$。

(b) 不妨假设 F 表示答错同学的比例。那么权重的变化为

$$\beta(Fw)+(1-F)w=(1-(1-\beta)F)w.$$

与 (a) 类似的，假设询问 k 次，每次答错同学的比例记为 F_i，那么最终总权重会有上界：

$$(1-(1-\beta)F_k)\,(1-(1-\beta)F_{k-1})\cdots(1-(1-\beta)F_1)\,n.$$

注意到

$$1-(1-\beta)F_i \leq e^{-(1-\beta)F_i},$$

则有

$$\beta^m \leq w \leq e^{-(1-\beta)\sum_i F_i} \cdot n.$$

由于 F_i 实际就是算法在第 i 次判断错误的概率，所有判断错误的期望就是 $\sum_i F_i$。

则 $\sum_i F_i \leq \frac{m \ln 1/\beta + n}{1-\beta}$。