These solutions are **unofficial**. Please let me know if there are any errors in this document.

1. (a) True. $n^2 \leq n \lg n + n^2 \leq 2 \cdot n^2$ for $n > 2$.
   (b) False. $5^{\lg \lg n} = (\lg n)^{\log_2 5} = \omega((\lg n)^2)$.
   (c) False. $(2n)! > n^n$ for all $n > 1$.
   (d) False. P is a subset of NP.
   (e) True. Both 3-SAT and SET COVER are in NP-Complete.
   (f) True. KNAPSACK is in NP-Complete.

2. (a) $T(n) = \Theta(n^2 \lg^2 n)$ by Case 2 of Master's Theorem.
   (b) $T(n) = \Theta(n)$ by substitution method.
   (c) $T(n) = \Theta(\lg \lg n)$ by recursion tree.
   (d) $T(n) = \Theta(n^2)$ by substitution method / recursion tree.
   (e) $\Theta(n)$. By the aggregate method, the total cost of n operations $\leq (2^0)^2 + (2^1)^2 + \cdots + (2^{\lfloor \lg n \rfloor})^2 + 3n \leq 2 \cdot 2^{2\lfloor \lg n \rfloor} + 3n \leq 2n^2 + 3n$. At the same time, the total cost of n operations $\geq (2^{\lfloor \lg n \rfloor})^2 \geq (\frac{n}{2})^2$. Hence, the total cost of n operations is $\Theta(n^2)$ and the amortized cost of each operation is $\Theta(n)$.

3. The algorithm should be as follows. It runs in $\Theta(n \log n)$ due to sorting.

   - Sort w and r both in ascending order.
   - Match w_i to r_i for $i \in \{1, 2, \ldots, n\}$.
   - The total payment is $\sum_{i=1}^{n} w_i / r_i$.

   Optimal substructure property: Assume without loss of generality that w_1 is matched to r_1 in the optimal solution, then $\text{MINCOST}(w, r) = \text{MINCOST}(w \setminus w_1, r \setminus r_1) + w_1/r_1$.

   Proof. Consider an optimal solution OPT. Since $\texttt{OPT} \setminus (w_1, r_1)$ is a valid solution to the subproblem $(w \setminus w_1, r \setminus r_1)$, we have $\text{MINCOST}(w, r) \leq \text{MINCOST}(w \setminus w_1, r \setminus r_1) + w_1/r_1$.

   Suppose $\text{MINCOST}(w, r) > \text{MINCOST}(w \setminus w_1, r \setminus r_1) + w_1/r_1$. We can take the optimal solution OPT1 to $(w \setminus w_1, r \setminus r_1)$. Then, $\texttt{OPT1} \cup (w_1, r_1)$ forms a better solution to the original problem with cost $< \text{MINCOST}(w, r)$, contradiction.

   Greedy choice property: Suppose w and r are sorted in ascending order. Then, w_1 is paired with r_1 in some optimal solution.

   Proof. Consider any optimal solution OPT. If w_1 is paired with r_1, we are done. Otherwise, let (w_1, r_j) and (w_i, r_1) be the pairs involving w_1 and r_1. We create a new solution by pairing (w_1, r_1) and (w_i, r_j) instead. We have

   $$\frac{w_1}{r_1} + \frac{w_i}{r_j} = \frac{w_1 r_j + w_i r_1}{r_1 r_j} \leq \frac{w_1 r_1 + w_i r_j}{r_1 r_j} = \frac{w_1}{r_j} + \frac{w_i}{r_1}$$

   and hence the new solution is no worse than OPT.

4. (a) FABULOUSHALF is in NP since we can use the fabulous set S (of size exactly $n/2$) as the certificate, and construct a verifier that checks whether $\sum_{i \in S} a_i \geq A/2$ and $\sum_{i \in S} b_i \geq B/2$ in $O(n)$ time, which is polynomial in n.

   To show that FABULOUSHALF is NP-hard, we show a reduction from PARTITIONEQUAL to FABULOUSHALF:

   Let x be an input instance to PARTITIONEQUAL. We construct an input instance to FABULOUSHALF by taking $a_i = x_i$ and $b_i = \max\{x_1, \ldots, x_n\} - x_i$. Given that x_i are non-negative integers for all i, a_i and b_i are also non-negative integers. Next, we show that the instance to PARTITIONEQUAL is a YES-instance if and only if the instance to FABULOUSHALF is a YES-instance.

   - ($\Rightarrow$) There exists $T \subseteq \{1, 2, \ldots, n\}$ such that $|T| = n/2$ and $\sum_{i \in T} x_i = \sum_{i \notin T} x_i$. Then, $\sum_{i \in T} x_i = \sum_i x_i/2$. Take $S = T$, we have $\sum_{i \in S} a_i = \sum_{i \in T} x_i = \sum_i x_i/2 = \sum_i a_i/2$ and $\sum_{i \in S} b_i = (n/2)\max\{x_1, \ldots, x_n\} - \sum_{i \in T} x_i = (n/2)\max\{x_1, \ldots, x_n\} - \sum_i x_i/2 = \sum_i b_i/2$. Hence (a, b) is a YES-instance to FABULOUSHALF.
   - ($\Leftarrow$) There exists $S \subseteq \{1, 2, \ldots, n\}$ such that $|S| = n/2$, $\sum_{i \in S} a_i \geq A/2$ and $\sum_{i \in S} b_i \geq B/2$. Taking $T = S$, we have $\sum_{i \in T} x_i = \sum_{i \in S} a_i \geq \sum_i a_i/2 = \sum_i x_i/2$. Also, since $\sum_{i \in S} b_i \geq B/2$, we have $(n/2)\max\{x_1, \ldots, x_n\} - \sum_{i \in S} x_i \geq (n/2)\max\{x_1, \ldots, x_n\} - \sum_i x_i/2$, which implies $\sum_{i \in T} x_i \leq \sum_i x_i/2$. Combining both conditions, we have $\sum_{i \in T} x_i = \sum_i x_i/2$ and x is a YES-instance to PARTITIONEQUAL.

(b) For integers $p \in \{0, 1, \ldots, A\}, q \in \{0, 1, \ldots, B\}$, $s \in \{0, 1, \ldots, n/2\}$ and $k \in \{0, 1, \ldots, n\}$, let $m[p, q, s, k]$ be whether there exists a set $S \subseteq \{1, 2, \ldots, k\}$ such that $|S| = s$, $\sum_{i \in S} a_i = p$ and $\sum_{i \in S} b_i = q$. We have the following recurrence:

$$m[p,q,s,k] = \begin{cases} 1 & \text{if } k = 0 \text{ and } (p,q,s) = (0,0,0) \\ 0 & \text{if } k = 0 \text{ and } (p,q,s) \neq (0,0,0) \\ \max\{m[p,q,s,k-1], m[p-a_k, q-b_k, s-1, k-1]\} & \text{if } k > 0, p \geq a_k, q \geq b_k \\ m[p,q,s,k-1] & \text{otherwise} \end{cases}$$

Here, if $k = 0$ then $(p, q, s) = (0, 0, 0)$ is clearly the only achievable pair. Else, if $k > 0$, we try checking whether using the k-th sheet of paper or using only the first $k - 1$ sheets leads to a feasible set. To get the final answer, we check whether any entry $m[p, q, n/2, n]$ is 1, where $p \geq A/2$ and $q \geq B/2$.

The running time is $O(n^2AB)$. To achieve this running time, we fill in the table in the order $m[*, *, *, 0], m[*, *, *, 1], \ldots, m[*, *, *, n]$. Since the value of $m[p, q, s, k]$ depends only on $m[*, *, *, k - 1]$, we have all necessary values to compute $m[p, q, s, k]$. There are $A \cdot B \cdot n \cdot n$ entries, and computing each entry takes time $O(1)$. Computing the final answer takes time $O(AB)$. Thus, the overall running time is $O(n^2AB)$.

(c) The size of the input for this problem is $O(n(\log A + \log B))$. However, the algorithm in (b) is polynomial in A and B, which is considered exponential in the size of the input. Hence, this is not a polynomial time algorithm (with respect the input size).