Problem 1 (Change Money) Consider a sequence of n positive integers $a_1, a_2, \ldots, a_n$ which are values of n kinds of coins. The sequence is required to satisfy:

(1) $a_1 = 1$, and

(2) $a_i < a_{i+1}$ for all $1 \le i < n$.

Let N be a positive integer and consider the following greedy strategy to change N using $a_1, a_2, \ldots, a_n$:

> "Always use the coin with largest possible value"

Give a sufficient and necessary condition for the sequence $a_1, a_2, \ldots, a_n$ such that the greedy strategy above results in a way of changing N with minimum number of coins. Your condition are required to verifable in polynomial time.

Problem 2 (Matroid and Greed) Let $(V, \mathcal{I})$ be an independent system[1] and $c : V \to \mathbb{R}^+$ a cost function. Consider the following algorithm:

```
Greedy Algorithm

Input:   (V, I) and c : V -> R
Output:  A set S ∈ I
 1 Let V = {a_1, a_2, ..., a_n} be an enumeration of elements in V such that c(a_i) >= c(a_{i+1})
   for all 1 <= i < n.
 2 S = ∅
 3 For i = 1 to n
   3.1 If S ∪ {a_i} ∈ I then S = S ∪ {a_i}.
 4 Output S.
```

Show that $(V, \mathcal{I})$ is a matroid[2] *if and only if* the above algorithm outputs a $S \in \mathcal{I}$ with maximum cost for any cost function c.

Problem 3 (Matroid and Greed, Online Version) Let $(V, \mathcal{I})$ be a matroid and $c : V \to \mathbb{R}^+$ a cost function. Consider the following algorithm:

[1] The definition of independent system is the following: $\mathcal{I} \subseteq 2^V$ is a family of subsets of V. It is required to satisfy for every $S \in \mathcal{I}$, if $T \subseteq S$, then $T \in \mathcal{I}$

[2] An independent system $(V, \mathcal{I})$ is a matroid if it satisfies the following exchange property: If $S, T \in \mathcal{I}$ and $|S| < |T|$ then $S \cup t \in \mathcal{I}$ for some $t \in T \backslash S$.

Online Greedy Algorithm

Input: $(V, \mathcal{I})$ and $c : V \to \mathbb{R}$
Output: A set $S \in \mathcal{I}$

1 Let $V = \{a_1, a_2, \ldots, a_n\}$ be an enumeration of elements in V.

2 $S = \varnothing$.

3 For $i = 1$ to n

3.1 If $S \cup \{a_i\} \in \mathcal{I}$ then $S = S \cup \{a_i\}$ else

3.2.1 Let $J = \{j \mid S \backslash \{a_j\} \cup \{a_i\} \in \mathcal{I}, a_j \in S\}$ and $j^* = \arg\min_{j \in J} c(a_j)$.

3.2.3 If $c(a_{j^*}) < c(a_i)$ then $S = S \backslash \{a_{j^*}\} \cup \{a_i\}$.

4 Output S.

Show that the above algorithm outputs the $S \in \mathcal{I}$ with maximum cost.