13. The Set Packing problem is defined as follows: Given a family of sets $S_1, \ldots, S_m \subseteq U$ such that, for $1 \leq i \leq m$, $|S_i| = 3$ and has profit $c(S_i)$, find a subset of these sets that maximizes the profit, while each element is covered at most once.

Q1. Provide a integer programming formulation for the problem (IP-SC). For this consider m variables $x_1, \ldots x_m$ where variable x_i indicates whether set S_i is or is not in the packing.

Let LP-SC be the Linear Programming problem obtained by relaxing the $x_i \in \{0,1\}$ conditions in IP-SC by $x_i \in [0,1]$.

Consider the following algorithm that, first computes an optimal solution x^* to LP-SC, and second rounds the solution according to the the following randomized algorithm:

- (1) Choose a set S_i to be in the solution with probability $x_i^*/6$.
- (2) If an element $u \in U$ is covered by more than one set, remove all the sets in the solution that contain u.

Q2. Show that the proposed algorithm is a randomized 12-approximation for Set packing

Maximizar:
Σi=1 a m c(Si) * xi

Sujeto a:
xi ∈ {0,1}, para 1 ≤ i ≤ m
Σi=1 a m xi * I(e ∈ Si) ≤ 1, para todo e ∈ U

donde:

xi es una variable binaria que indica si se incluye o no el conjunto Si en el empaquetamiento. c(Si) es el beneficio asociado a incluir el conjunto Si en el empaquetamiento. U es el conjunto universal que contiene todos los elementos de todos los conjuntos Si.I(e ∈ Si) es una variable indicadora que toma el valor 1 si el elemento e está en el conjunto Si y 0 en caso contrario.

La función objetivo pretende maximizar el beneficio total de los conjuntos seleccionados, mientras que las restricciones garantizan que cada elemento esté cubierto como máximo una vez por los conjuntos seleccionados. En concreto, la segunda restricción garantiza que la suma de xi de todos los conjuntos que contienen un elemento e dado sea como máximo 1. Si se selecciona un conjunto Si (es decir, xi=1), entonces contribuye a la cobertura de todos sus elementos, y por tanto la suma Σi=1 a m xi * I(e ∈ Si) debe ser menor o igual que 1.