3. *Una pila de pedretes.* Considereu el joc següent. Tenim una pila de n pedretes i som dos jugadors. A cada moviment podem treure de la pila un cert nombre de pedretes. Cadascun de nosaltres tenim assignat un conjunt $S \subseteq \{1, ..., n\}$ i $T \subseteq \{1, ..., n\}$, respectivament, indicant el nombre de pedretes que podem treure en el nostre torn. Per exemple, si el meu conjunt és $S = \{1, 2, 3\}$ i el teu conjunt és $T = \{2, 4, 6\}$, en el meu torn només puc treure 1, 2 o 3 pedretes de la pila i en el teu torn tu pots treure 2 o 4 o 6 pedretes de la pila. No podem treure de la pila més pedretes de les que hi ha i qui primer buidi la pila guanya. Si la pila té menys pedretes de les que pot treure el jugador a qui li toca jugar, és a dir no té cap jugada possible, aleshores considerem empat. Per exemple si $n = 8$ i és el meu torn, puc guanyar treient 3 pedretes. Queden 5 pedretes: tu en pots treure 2 o 4, deixant 3 o 1 pedretes a la pila, respectivament. I en cada cas jo puc buidar la pila.

   Definim el el problema BuidarPila:

   > Donats n, $S \subseteq \{1, ..., n\}$ i $T \subseteq \{1, ..., n\}$,
   > decidir si el jugador amb S que inicia el joc té una estratègia guanyadora (pot guanyar el joc).

   Classifiqueu la complexitat computacional d'aquest problema.

Per resoldre aquest problema construim un graf dirigit i bipartit G = (V,E) que relaciona l'estat de la partida en el torn de un jugador amb els possibles estats als que pot arribar amb un moviment. L'estat esta definit per el numero de pedretes restants i el jugador al que li toca.

V = tots els possibles estats de la parida (2n): {S_0,S_1,...S_n} U {T_0,T_1,...,T_n}

E = relaciona cada estat amb un altre estat accessible amb un sol moviment:

- (S_x,T_y) € E si x - mov = y, mov € S
- (T_x,S_y) € E si x - mov = y, mov € T

On S i T són els moviments possibles de cada jugador.

Ara definim 3 etiquetes pels estats:

- V victoria
- D derrota
- E empat

Iniciem definint: S_0 = D, T_0 = V, tots els altres = E i extenem les etiquetes d'aquesta manera:

Iterem sobre els vèrtex en ordre creixent de la mida de la pila (ja que la etiqueta d'un node només depèn dels que tenen una pila més petita).

- Si S_i pot accedir a un node T_j tal que T_j = V -> S_i = V
- Si S_i només pot accedir a nodes T_j tal que T_j = D -> S_i = D
- Si T_j només pot accedir a nodes S_i tal que S_i = V -> T_j = V
- Si T_j pot accedir a un node S_i tal que S_i = D -> T_j = D

.

Si S_n = V tenim una estratègia guanyadora, sino no.

Cost:

Recorrem els 2n vèrtex i el cost de mirar tots els adjacents a cada un és O(n), per tant l'algorisme té cost O(n^2).

Per emmagatzemar el graf necessitem $O(n^2)$, ja que tenim $2n$ vèrtex i $O(n^2)$ arestes.