Lab05: Subset_Sum

1. **Decidir si hay una subcolección de elementos que sumen una cantidad**

   En este ejercicio dispones del fichero (`lab05_subset_sum.py`), donde tienes que implementar la función `has_sum`. La función `has_sum`, dado un valor ≥ 0 y una colección de positivos, devuelve `True` si existe una subcolección que sume el valor de entrada. En otro caso, devuelve `False`.

   Para probar la función debes eliminar los comentarios de todas las instrucciones `assert` correspondientes a `has_sum` que se encuentran en `test()`.

2. **Encontrar una subcolección de elementos que sumen una cantidad**

   Ahora tienes que implementar la función `subset`, que dado un valor ≥ 0 y una colección de positivos, si existe una subcolección que sume el valor de entrada, la devuelve. En otro caso, devuelve la lista [`None`].

   Para probar la función debes eliminar los comentarios de todas las instrucciones `assert` correspondientes a `subset` que se encuentran en `test()`.

3. **Medir el tiempo de ejecución**

   Podemos medir el tiempo transcurrido durante la ejecución de una determinada función, a través del módulo `timeit`. Lo usaremos en el test de este laboratorio para calcular el tiempo de ejecución de tu función `subset`.

   Comenta todos los `assert` y descomenta la colección 6 junto con el `assert subset`. Cuando ejecutes el test, apunta el tiempo transcurrido. Añade un comentario a tu función `subset` indicando el tiempo que ha tardado la ejecución de la colección 6.

4. **Optativo:** Implementa la función `all_subsets` que debe encontrar todas las subcolecciones que sumen una cantidad.

   La función `all_subsets` devuelve una lista con todas las subcolecciones que sumen una cantidad o bien la lista [`None`].