Práctica 3: Cifrado por Mochilas y Mochilas trampa

1 Se deberán crear las funciones letter2ascii y ascii2letter que permita hacer la siguiente traducción (si existe en python, pueden utilizarse las que ya estén definidas):

A	65	0100 0001
B	66	0100 0010
C	67	0100 0011
D	68	0100 0100
E	69	0100 0101
F	70	0100 0110
G	71	0100 0111
H	72	0100 1000
I	73	0100 1001
J	74	0100 1010
K	75	0100 1011
L	76	0100 1100
M	77	0100 1101

N	78	0100 1110
O	79	0100 1111
P	80	0101 0000
Q	81	0101 0001
R	82	0101 0010
S	83	0101 0011
T	84	0101 0100
U	85	0101 0101
V	86	0101 0110
W	87	0101 0111
X	88	0101 1000
Y	89	0101 1001
Z	90	0101 1010

De igual forma, deberán crearse programas que permitan la reconstrucción del mensaje a posteriori (cuando tengamos múltiples letras).

2 Funciones para mochilas.

- Crear una función **knapsack** que tome un vector fila y determine si es una mochila supercreciente (devolviendo 1), una mochila no supercreciente (devolviendo 0) o no es una mochila (devolviendo -1).
- Crear una función **knapsacksol** que tome una mochila (supercreciente o no) s, un valor v, y determine usando el algoritmo de mochilas supercrecientes si v es valor objetivo de s.
- Crear la función **knapsackcipher** que cifra un mensaje a partir de una mochila (no necesariamente creciente). Deberá tomar el texto, la mochila y devolver un vector numérico con el mensaje cifrado.
- De igual modo, crear **knapsackdecipher** que descifra un vector numérico conocida la mochila supercreciente asociada.

3 Funciones para mochilas trampa.

- Función **commonfactors**, que toma un valow w y una mochila s, y comprueba si w tiene factores primos comunes con alguno de los elementos de s. (Nota: No nos interesa tomar w si los hay).
- Función **knapsackpublicandprivate**, que genera una pareja de claves pública (mochila con trampa) y privada (w y m) a partir de una mochila supercreciente. Observaciones: (i) se deberá pedir al usuario que elija el valor de m (que se usará para tomar el módulo) y comprobará que el número escogido es adecuado, (ii) para la búsqueda de w se debe dar la opción al usuario de hacer la búsqueda de forma aleatoria o bien en un rango dado (o los dos). El valor de w debe ser dado de forma que sea invertible módulo m y que no tenga primos comunes con los elementos de s.

- Función **knapsackdeciphermh** que toma la cadena supercreciente, los elementos m y w y un criptograma (vector numérico) y devuelve el mensaje descifrado.

4 Finalmente implementar el método de criptoanálisis de Shamir y Zippel. El programa debe realizarse para que trabaje en distintos rangos, solicite al usuario (en caso de que en un primer rango no encuentre solución) si quiere seguir con el rango siguiente e indicar el tiempo requerido en cada paso (es decir, cuanto ha tardado en un rango dado, encuentre o no solución). En caso de encontrar solución, deberá devolverse adicionalmente la mochila supercreciente asociada.

Recordad que todo el código realizado deberá estar convenientemente documentado.